

PROgrammez !

www.programmez.com

Mensuel n°159 - janvier 2013



Choisir sa plateforme de développement mobile

Quel OS pour quelles apps ?
Les outils, les avis de développeurs.
Nos conseils.

Développeur mobile
une carrière en plein boom ?

● C++

Découvrir la
programmation C++ AMP

● Objet connecté

Le proj & Koneki

● Webmaster

No.de. JS : JavaScript, côté serveur
Personnaliser Prestashop

● Big Data

Comprendre Hadoop

Printed in EU - Imprimé en UE - BELGIQUE 6,45 €
SUISSE 12 FS - LUXEMBOURG 6,45 € DOM Surf 6,90 €
Canada 8,95 \$ CAN - TOM 9,40 XPF - MAROC 50 DH

M 04319 - 159 - F: 5,95 €



DÉVELOPPEZ 10 FOIS PLUS VITE

WINDEV®

ATELIER DE
GÉNIE LOGICIEL
PROFESSIONNEL



DÉVELOPPEMENT D'APPLI POUR TABLETTE*: RECOMPILÉZ VOTRE APPLICATION WINDOWS ACTUELLE, ET HOP, ÇA MARCHE**

VERSION
EXPRESS
GRATUITE
Téléchargez-la !

* IPAD, ANDROID, WINDOWS 8 & RT



** : L'APPLICATION WINDOWS DOIT SIMPLEMENT AVOIR ÉTÉ
DÉVELOPPÉE AVEC LE NUMÉRO 1 DES AGL EN FRANCE, WINDEV.
VOUS ÉCONOMISEZ DES MOIS DE DÉVELOPPEMENT

(il faudra peut-être retailer quelques fenêtres...)

Et ça marche également avec les applis Mac et Linux
développées avec WINDEV.



« WINDEV 18 »

Atelier de développement
multi-plateformes: Windows 8,
RT, 7, Vista, XP..., Linux, Internet,
Android, iPhone, iPad...

WINDEV Mobile 18 compile les applications mobiles.
WINDEV 18 compile les applications Windows, Mac et Linux.
WEBDEV 18 crée les sites Internet et Intranet.

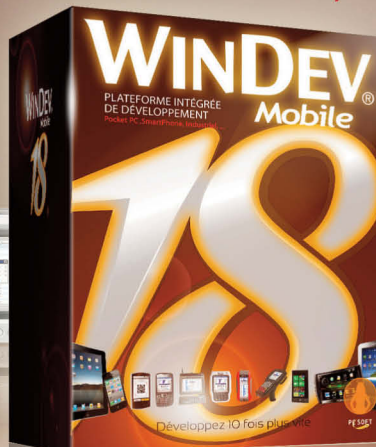
Votre code source est unique

**DEMANDEZ VOTRE
DOSSIER GRATUIT**

260 pages - 100 témoignages - DVD
Tél: 04.67.032.032 info@pcsoft.fr

www.pcsoft.fr

Fournisseur Officiel de la
Préparation Olympique



Les objets connectés, internet des objets : **nouvel Eldorado** technologique ?

LeWeb 2012 (4-6 décembre) était placé sous le signe de l'internet des objets, ou plus concrètement de ce que l'on appelle les objets connectés, grosso-modo, le M2M (Machine to Machine). Le marché M2M va de l'automobile aux appareils ménagers. En 2012, plus de 140 millions de modules M2M se sont vendus (chiffre IDATE), mais le marché va croître rapidement : + 30 % en 2015. L'Europe est une grande consommatrice de M2M, surtout dans le secteur automobile.

L'objet connecté est un appareil qui peut se connecter à un réseau, interagir et échanger avec d'autres appareils, sans qu'il y ait besoin de passer, de transiter par un ordinateur. Ces objets sont donc des entités technologiques indépendantes.

Un des exemples les plus remarquables durant LeWeb est l'étonnant thermostat Nest, objet que l'on avait déjà vu il y a plus d'un an en démonstration. L'objet se veut intelligent dans le sens où il apprend de vos habitudes et de son environnement pour ajuster au mieux la chaleur, optimiser la consommation électrique. Il récupère les informations météo. Le Nest peut résumer le concept de l'objet connecté : l'usage et le design.

Tous les objets sont concernés. Nous pouvons prendre l'exemple du pèse-personne de Withings, société française. La **balance wifi** facilite le suivi de chaque personne de la famille pour le poids, avec estimation de la proportion de muscle et de gras, etc. On peut créer un programme sportif pour améliorer votre forme et le profil de masse corporelle. Le tout est relié à des applications sur son smartphone ou son ordinateur. Le constructeur propose aussi un tensiomètre capable d'établir le suivi de la tension, de fournir tous les chiffres clés, etc. Withings cherche à trouver des usages motivants



pour l'utilisateur. L'objet connecté ne doit pas être un gadget technologique supplémentaire mais s'intégrer au quotidien et devenir naturel. Beaucoup d'objets connectés de loisir étaient visibles à LeWeb.

> Mon (très) précieux

Finalement, est-ce une véritable révolution ou une simple évolution ? Pour le moment, c'est la 2e voie qui est prise. Le cas des **ampoules LIFX** en est un bel exemple. Il s'agit d'ampoules connectables à un réseau wifi que l'on peut ensuite piloter depuis un smartphone, une tablette. Le but : personnaliser, repenser le fonctionnement des douilles électriques (fini l'interrupteur !). L'idée est intéressante et plus simple que la domotique classique. On dispose d'un SDK de développement pour créer des usages. Le principal problème est son prix : 69 \$ pièce ! Il faudra plusieurs années avant que l'objet connecté se définisse mieux et s'intègre à l'usage au quotidien et surtout inventer de nouveaux usages.

Nous pouvons aussi regarder vers d'autres

objets ou plutôt de nouveaux modes d'intégration entre objets. Le projet de Microsoft, SmartGlass, est intéressant à regarder. L'objectif vise à partager tous les contenus entre les différents terminaux disponibles : tablette, smartphone, Xbox / Kinect sur le téléviseur. Mais il s'agit aussi d'interagir avec un contenu, un jeu, avoir des compléments. Là, tout reste à faire et le développeur peut s'éclater avec le SDK SmartGlass.

Nous avons déjà évoqué les lunettes Google (Project Glass). En quelque sorte, il s'agit de proposer une réalité augmentée, avec des lunettes Android et des connexions, pour fournir informations, conseils, reconnaître des amis, navigation, etc. Là encore, il est nécessaire que la technologie mûrisse.

Les développeurs auront un rôle à jouer car sans eux, pas d'applications, pas d'interaction. Mais le développeur ne sera pas seul, l'objet connecté nécessite un design particulier (aussi physique que virtuel), des usages définis et du contenu. Ce triptyque nous semble incontournable.

Pour conclure, parlons d'un autre projet vu à LeWeb : SmartThings. Le but est de créer une plateforme sur laquelle on connecte

l'ensemble des appareils, terminaux de la maison. Pour cela, on utilise des capteurs connectés. Bref, une

autre forme de domotique qui se veut plus simple. Pour l'éditeur, 3 choses sont essentielles : connecter les objets de tous les jours à Internet / à un réseau, avoir une application mobile pour contrôler, gérer et surveiller ces objets, et enfin, avoir des outils simples pour développer les usages, les interactions. Là encore, le développeur est au cœur des préoccupations. Le kit SmartThings s'articule autour d'un hub et d'un ensemble de capteurs.

Les vidéos LeWeb 2012 : <http://www.youtube.com/user/leweb>

“Une domotique repensée”



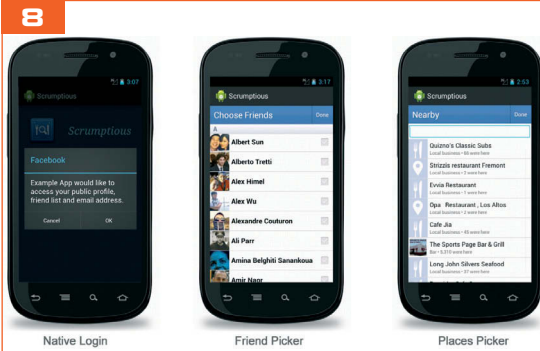
François Tonic

BUZZWORD

Les objets connectés, internet des objets : nouvel Eldorado technologique ?3

ACTUS

En bref6
Retour sur les Epitech Innovative Project 20129
C++ML : programmation visuelle en C++11

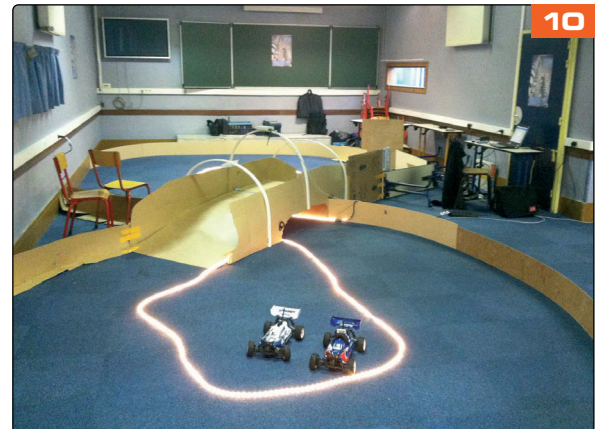


RESSOURCES

Communautés et agenda du mois12

DÉVELOPPEUR DU MOIS

Samuel Blanchard :
Les multiples vies d'un geek-développeur16



MATÉRIEL

Sécuriser et protéger
votre matériel informatique18

OUTILS

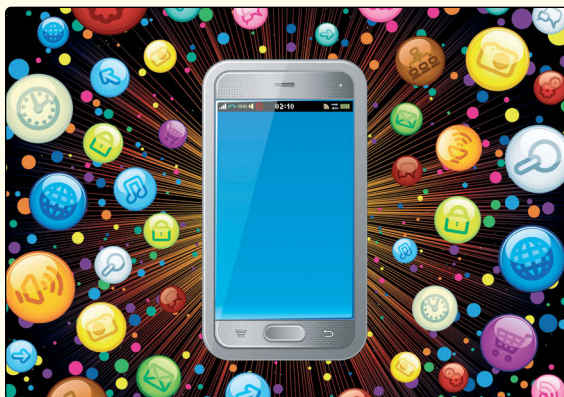
Meetup SkySQL à Paris :
Cloud, MySQL, MariaDB19
Neo4j : l'autre univers du stockage20

VEILLE TECHNO

Analysez vos données
en Map Reduce avec Hadoop !26

DOSSIER : ANDROID, IOS, WINDOWS PHONE 8

Choisir sa plateforme de développement mobile



Quels critères pour quelle plateforme ?31
Android, iOS, Windows Phone 8 :
les développeurs
parlent aux développeurs.....33
5 leçons mobiles à méditer40
La dictature des Public Stores
est-elle légitime ?45
Focus sur Windows Phone 846

- Développer pour **Facebook**
La plateforme, les API et SDK, opengraph, le modèle économique
- **Cloud, open Cloud** : ce qui change pour les développeurs

CARRIÈRE

Développeur mobile :
une carrière
pleine d'avenir ?50

**JE DÉBUTE
AVEC...**

Migrez vos applications
vers ASP.NET 4.5 (2e partie).....51

Je débute avec le kit
de démarrage ASP.net MVC.....53

Découverte de C++AMP
avec Visual Studio 201258

PRATIQUE

Koneki, le développement M2M simplifié.....67

WEBMASTER

JavaScript :
la révolution est-elle en marche ?
(2e partie)

Créer un service Cloud en moins de 500 lignes70

Implémentation d'applications Web
avec Node et Express73

JavaScript/développeur :
l'Amour avec un grand A.....76

JavaScript est souvent incompris78

Ecrire un module d'extension à Prestashop79



18



©F. Tonic

**5.1 c'est mieux que 5.0,
ainsi parle le W3C**

J'ai d'abord cru à une bonne blague : lancement des premiers brouillons de HTML 5.1. « Le 17 décembre, le W3C annonce aujourd'hui la publication de la définition complète de HTML 5 et des spécifications du Canvas2D ». Pas de doute, il s'agissait bien d'une annonce officielle.

Mais l'affaire est un peu plus compliquée que cela. Bien que la définition complète de HTML 5 soit fixée, il reste maintenant à proposer des batteries de tests et d'interopérabilité et surtout à sortir la recommandation finale W3C officielle du HTML 5. Cette phase interviendra seulement en 2014 (et officiellement, pas avant le dernier trimestre). Soit dans environ 18 mois. Les développeurs peuvent maintenant examiner les spécifications HTML 5 et estimer les modifications à apporter. Surtout, les éditeurs pourront implémenter les fonctions manquantes de HTML 5, corriger les mécanismes existants. On peut s'attendre à de nombreuses mises à jour des navigateurs et des moteurs de rendu HTML. Les environnements de développement devront aussi être corrigés. Il reste encore un important travail de corrections de bugs (169 trouvés au 17 décembre). La question de la sécurité et certains mécanismes ne manqueront pas de remonter à la surface.

Pour ne pas perdre de temps, le W3C engage dès maintenant les premiers travaux autour de HTML 5.1. Une interopérabilité totale sera assurée avec la 5.0 (ouf !), ainsi que la correction des problèmes de la version actuelle. Il est prévu que cette évolution inclue moins de technologies. Dans le même temps, le consortium lance le travail sur Canvas 2D, niveau 2.

Il s'agit encore d'une toute première investigation autour de la 5.1. Il faudra attendre 2014 pour voir apparaître les premières spécifications concrètes. Les recommandations officielles ne sont pas attendues avant fin 2016 ! Et dans les plans du W3C, il est déjà prévu de lancer les réflexions autour de la 5.2 dès 2015...

Est-ce là un moyen pour le W3C de couper les ambitions du WHATWG qui avait créé un fork de HTML 5 l'été dernier ? Si rien n'est dit officiellement, c'est effectivement un moyen d'accélérer, un peu, le calendrier futur.

HTML 5, associé à CSS et JavaScript, est aujourd'hui un modèle de développement central pour plusieurs plateformes desktop et mobiles. Microsoft mise beaucoup dessus, Adobe aussi, tout comme Apple et Google.

Pour ce dernier numéro de l'année, nous allons plonger dans l'univers impitoyable des smartphones et des tablettes. Des développeurs témoignent de leur expérience de développement et vous donnent des conseils pratiques. Aujourd'hui, comme vous le verrez, il y a deux leaders : iOS et Android. Windows Phone 8 ambitionne d'être la troisième plateforme mobile mais ce n'est qu'avec une large diffusion que WP8 pourra prétendre rivaliser avec la concurrence.

En tout cas, il faudra le faire vite, car BlackBerry 10 arrivera dans quelques semaines et espère bien relancer la machine. Au-delà du système et du nombre d'applications c'est clairement la masse d'utilisateurs qui joue.

Vous découvrirez les joies du développement C++ avec les nouvelles librairies AMP ainsi qu'une introduction à Hadoop et aux objets connectés...

François Tonic - Rédacteur en chef
ftonic@programmez.com



**Toute l'équipe de PROGRAMMEZ !
vous souhaite
une excellente année 2013.**

Editeur : Go-02 sarl, 21 rue de Fécamp 75012 Paris -
diff@programmez.com. - Rédaction : redaction@programmez.com - Directeur de la Rédaction : Jean Kaminsky.
Rédacteur en Chef : François Tonic - ftonic@programmez.com. Ont collaboré à ce numéro : F. Mazué. Experts : J.C. Baudier, A. Picq, R. Poirson, S. Martins, N. Albeza, M. Cholley, R. Gay, L. Humez, P.Y. Le Borgne, J. Lecerf, T. Tanon, P.H. Muller, F.X. Bois, R. Saissy, H. Verreydt, J.B. Boisseau, R. Buclin, G. Clément, A. Sottiau, N. Tonneau, M. Herbaux, N. Suter, D. Tran, L. Jordi, B. Boucard, B. Cabé, M. Lux, T. Templier, M. Loussinian, M. Diaz-Orlich,

V. Saluzzo, M. Duvignaud, G. Brout, F. Muiseroux.

Illustration couverture : © istockphoto.com/piart

Publicité : Régie publicitaire, K-Now sarl.

Pour la publicité uniquement : Tél. : 01 41 77 16 03 -
diff@programmez.com. Dépôt légal : à parution -
Commission paritaire : 0717K78366 ISSN : 1627-0908.
Imprimeur : S.A. Corelio Nevada Printing, 30 allée de la
recherche, 1070 Bruxelles Belgique.

Directeur de la publication : J-C Vaudecrane

Abonnement : Programmez, 17, Chemin des Boulangers, 78926
Yvelines Cedex 9 - Tél. : 01 55 56 70 55 - abonnements.programmez
@groupe-gli.com - Fax : 01 55 56 70 91 - du lundi au jeudi de 9h30 à 12h30
et de 13h30 à 17h00, le vendredi de 9h00 à 12h00 et de 14h00 à
16h30. Tarifs abonnement (magazine seul) : 1 an - 11 numéros France
métropolitaine : 49 € - Etudiant : 39 € - CEE et Suisse : 55,82 € -
Algérie, Maroc, Tunisie : 59,89 € Canada : 68,36 € - Tom : 83,65 € -
Dom : 66,82 € - Autres pays : nous consulter. PDF : 30 € (Monde
Entier) souscription exclusivement sur www.programmez.com

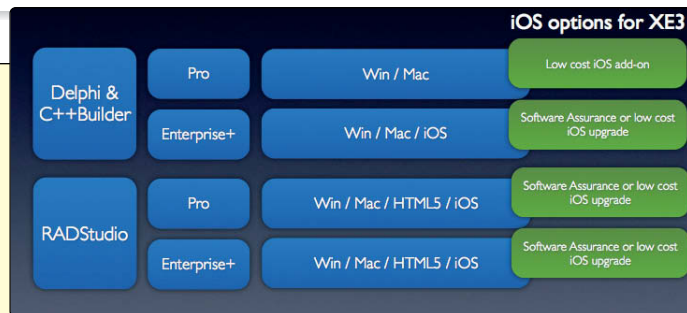
» C++ Builder XE 3 a été dévoilé par l'éditeur

Embarcadero le 12 décembre dernier. C++ Builder XE3 (développé à l'origine par Borland) est une plateforme complète de développement C++ multiplateforme pour les applications natives sur desktop : Windows, OS X, et bientôt pour iOS et Android. L'environnement intègre également un compilateur 64 bits complet et une mise à jour VCL pour les clients C++Builder existants, permettant une mise à jour 64 bits rapide pour des millions d'applications Windows existantes.

Embarcadero met en avant plusieurs fonctions et facilités :

- compilateur 64 bits
- utilisation du compilateur Clang 3.1 pour

supporter les plus récentes fonctions et bibliothèques C++ (particulièrement C++ 11) - support de Windows 8 et des derniers systèmes OS X ainsi que des écrans Retina (environnement Apple) grâce à une gestion plus fine du HiDPI. - les développeurs peuvent utiliser les extensions de standard C++ d'Embarcadero pour accélérer et rationaliser les développements C++. Les techniques agiles telles que le prototypage rapide, la programmation PME (Properties, Methods, Events) et le développement visuel permettent de développer jusqu'à 5 fois plus rapidement par rapport aux développements traditionnels.



Le développement mobile (Android / iOS) est attendu pour 2013. L'objectif est de supporter les fonctions matérielles et un large nombre d'API. Une première bêta devrait être disponible quand vous lirez ces lignes. Les modules iOS arriveront (estimation éditeur) première moitié de 2013 et vers la mi-2013 pour Android. Windows RT et Linux Server sont prévus mais à une date inconnue.

» Silverlight a-t-il encore un avenir ?

C'est la question qui se pose depuis de nombreux mois. La version 5 est sortie il y a plus d'un an et depuis, plus rien. Aucune annonce d'une version 6, aucune mise à jour importante de la v5, très peu de mention de la technologie quand Microsoft évoque Windows 8, Windows Phone 8. Une version 5.1 est sortie en mai 2012 pour corriger quelques bugs. Courant décembre, le site officiel silverlight.net a été purement et simplement fermé pour rediriger vers une sous-section de MSDN. Officiellement, le support de Silverlight 5 s'étend jusqu'en 2021. Le modèle de développement web et web riche n'est plus du tout centré sur Silverlight.

» Amazon lance un outil de tests pour Android !

Amazon A/B est un outil gratuit pour tester et optimiser les applications Android. L'outil utilise des API Amazon (Insights SDK / API). Il fonctionne en ligne et est accessible aux développeurs inscrits au programme Mobile App Distribution. Il servira uniquement aux applications destinées à la boutique d'applications Amazon et non à Google Play. Site : <https://developer.amazon.com/sdk/ab-testing.html>

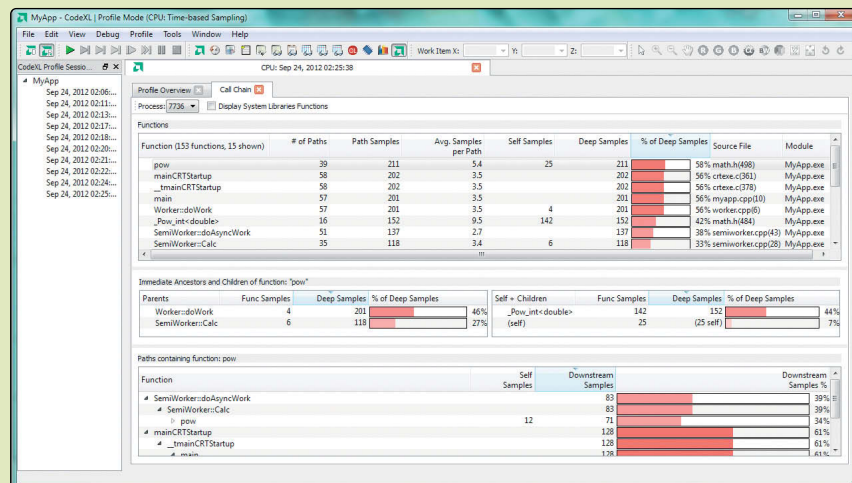
» Twitter décide d'être plus transparent et réactif envers les développeurs.

Pour les aider à anticiper les changements et les nouvelles versions, Twitter propose maintenant « Calendar of API changes ». Toutes les évolutions jusqu'à mars 2013 sont indiquées avec les API impactées, les ressources et la disponibilité estimée.

Site : <https://dev.twitter.com/calendar>

» AMD livre une nouvelle version de APP SDK, la v2.8, avec une suite

d'outils adaptés, CodeXL. Cet ensemble de développement doit aider le développeur à optimiser et à utiliser des technologies parallèles et de CPU / GPU avec un seul code. Le SDK supporte OpenCL, C++, DirectCompute. Pour réaliser cette « indépendance » pour être le plus portable possible, AMD a introduit un nouveau template de librairie : Bolt. Celui-ci doit apporter un modèle de développement connu (très similaire au STL) et surtout une indépendance et une abstraction matérielle et technique. APP SDK fournit toutes les librairies nécessaires pour coder en OpenCL avec de tout nouveaux exemples de code. AMD avait été très critiqué sur son manque de réactivité vis-à-vis d'OpenCL, concurrent direct de CUDA. CodeXL 1.0 (disponible depuis début décembre) inclut un debugger GPU, un profileur CPU et GPU, un analyseur statique. Site : <http://developer.amd.com>



» Google Maps SDK, nouvelle version, est disponible pour iOS (iPhone / iPad).

Le retour de Maps sur la plateforme Apple s'est fait en force via une application dédiée et surtout avec la disponibilité d'un SDK spécifique. Il permet de personnaliser, interagir, d'afficher des cartes dans une application iOS. On dispose de toutes les fonctions essentielles (3D, rotation, zoom, etc.). Pour accéder au SDK, il faut envoyer une demande...

site : <https://developers.google.com/maps/documentation/ios/>



ComponentOne Studio Enterprise à partir de € 1 148



.NET Tools pour un développement intelligent : Windows, Web et XAML.

- Centaines de contrôles d'interface pour .NET : grilles, graphiques, rapports et planificateurs
- Support de Visual Studio 2012 et Windows 8
- Inclut Windows 8 Studios pour WinRT XAML et WinJS
- Nouveau thème Cosmopolitan (interface Windows 8) pour un aspect et ressenti moderne
- Déploiement et distribution sans droits

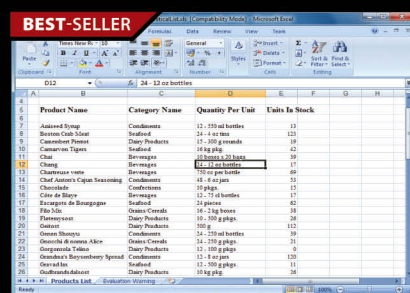


GdPicture.NET à partir de € 3 105



Outils complets d'imagerie documentaire et de gestion pour les développeurs.

- Capturez, traitez, créez, affichez, modifiez, annotez, composez, fractionnez, fusionnez et imprimez des documents depuis vos applications Windows et Web
- Lisez, écrivez et convertissez les images vectorielles et raster en plus de 90 formats, dont PDF, PDF/A, TIFF, GIF, JPEG, PNG, JBIG2, WMF, BMP, WBMP, ICO, PCX, PNM, XPM, JPEG 2000, HDR, PSD, TGA, PICT, EXR, DDS, PPM, SGI, PBM, PGM, PFM, XBM, IFF et le format graphique RAW

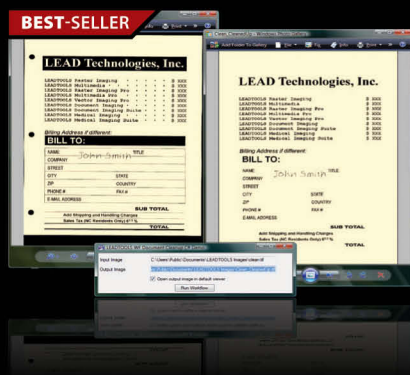


Aspose.Total for .NET à partir de € 1 940



Tous les composants Aspose .NET réunit dans un seul package.

- Programmez la gestion de formats courants, tels que Word, Excel, PowerPoint et PDF
- Incluez des graphiques, des e-mails, des correcteurs, des codes barres, l'OCR, des diagrammes, des images et la gestion de projets/formats dans les applications .NET
- Les principales utilisations sont la fusion de messages, l'ajout de codes barre, la création de rapports Excel dynamiques à la volée et l'extraction de texte des fichiers PDF



LEADTOOLS Document Imaging SDK V17.5 à partir de € 1 937

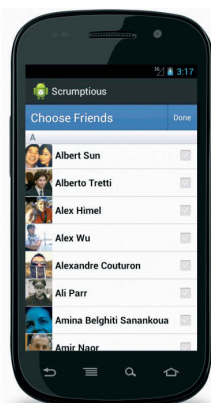


Incluez des fonctionnalités puissantes d'imagerie aux applications Windows, Web et mobile.

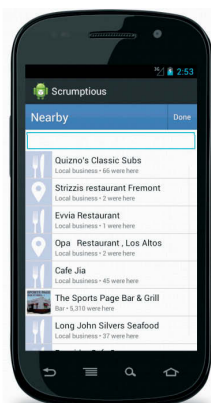
- Modules complémentaires d'OCR, OCR asiatique et arabe, ICR, OMR et MICR
- Modules codes barre disponibles : codes barre 1D et 2D
- Modules PDF disponibles : Lecture/écriture raster PDF et PDF avancé
- Lecteurs d'images fonctionnels à encombrement zéro et annotations pour tout périphérique supportant HTML5



Native Login



Friend Picker



Places Picker

» Les dernières nouvelles "développeurs" de Facebook...

Le réseau social Facebook fait évoluer régulièrement ses API, parfois, très brutalement. L'éditeur a annoncé la version 3 du SDK pour Android. Cette version est une version importante pour les développeurs : nouveaux objets d'interface, open graph, gestion des sessions simplifiée, API améliorées. On dispose aussi du Native Login (authentification sans sortir de l'application).

Le SDK pour iOS avait été mis à jour en octobre dernier, version 3.1 : intégration à iOS 6, Native Login, partage des posts... pour en savoir plus :

<https://developers.facebook.com>

» Micro Focus adapte Cobol à Windows 8.

Vous souhaitez développer en langage Cobol sur Windows 8 ? C'est possible avec la nouvelle version de Visual Cobol 2.1 de Micro Focus. Cette solution s'intègre à Visual Studio 2012 mais aussi à .Net et au Cloud. Les développeurs sont donc aujourd'hui en mesure de pérenniser leurs applications d'entreprise existantes sur la dernière version de la plateforme Microsoft. Vous pouvez ainsi développer, maintenir ou tout simplement migrer vos codes Cobol.

Site : <http://www.microfocus.com>

» SkySQL & Monty Program

annoncent la disponibilité de plusieurs connecteurs open source pour MySQL. Ils visent à faciliter la communication entre les applications C et Java et les bases MySQL et MariaDB. Il s'agit des composants : MariaDB Client Library for Java Applications et MariaDB Client Library for C pour la base de données MySQL. Ces connecteurs seront proposés à la communauté sous licence LGPL.

Site : <http://www.skysql.com/mariadb/connectors>

» PostgreSQL

a sorti début décembre, plusieurs mises à jour : Les nouvelles versions 9.2.2, 9.1.7, 9.0.11, 8.4.15 et 8.3.22. Ces versions corrigent plusieurs bugs, des problèmes d'index.

« Pour les utilisateurs de la PostgreSQL 9.2, il est recommandé de reconstruire les index créés avec la commande `CREATE INDEX CONCURRENTLY` à l'issue de la mise à jour. Cette opération est simple et n'a pas d'impact sur la production. Par ailleurs, l'arrêt du support de la branche 8.3 est annoncé pour février 2013. Si ce n'est pas déjà fait, il est temps de penser à migrer vers une ver-

» Google modifie sa plateforme cloud.

Google Cloud Platform arrive avec de nouveaux tarifs, de nouvelles fonctions. Cette plateforme inclut App Engine,



Cloud Storage, BigQuery, Compute Engine. Cloud Storage subit une baisse des prix de 20 % et l'éditeur lance une version de tests de Durable Reduced Availability, une option de stockage à prix réduit, sans les fonctions fines de répliquations. D'autre part, Google annonce des améliorations de ces datacenters européens. Et surtout, Compute Engine (un super IaaS) propose 36 nouvelles instances et une tarification revue à la baisse. Pour en savoir plus :

<https://cloud.google.com>

sion plus récente », précise Guillaume Lelarge, directeur technique de Dalibo. Détail des bug fix sur <http://www.postgresql.org/about/news/1430/>

» Google

a l'habitude de faire le ménage dans ses services et API plusieurs fois par an. Le ménage d'hiver, dévoilé mi-décembre, propose :

- l'arrêt de plusieurs fonctions de Google Calendar dès le 4 janvier 2013

- Google Sync va permettre l'accès à Gmail, Calendar et Contacts via CalDav et IMAP. Les utilisateurs actuels ne seront pas affectés par cette introduction.

- Google Calendar Sync, Google Sync for Nokia S60 et SyncML s'arrêtera en décembre 2013 et janvier 2014.

Pour en savoir plus : <http://googleblog.blogspot.ca/2012/12/winter-cleaning.html>

» Microsoft dévoile le futur d'ASP.Net.

Beaucoup de nouveautés sont attendues pour 2013 : OData, Windows Store, traçage, page d'aide, Windows Azure Active Directory, Facebook, templates mobiles, etc.

Les équipes en charge d'ASP.NET ratissent très large pour améliorer la plateforme. Cette version ne sera pas disponible avant fin 2013, avec une pré-version disponible vers septembre - octobre.

Pour tout savoir : <http://aspnetwebstack.codeplex.com/wikipage?title=Roadmap>

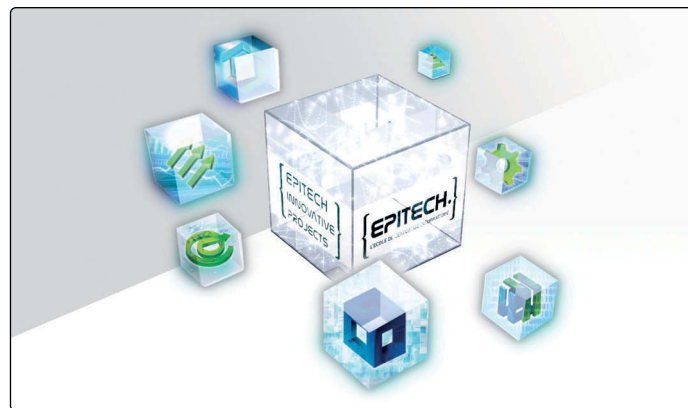
» Visual Studio 2012 update 1 est disponible depuis un mois.

Cette mise à jour cible le développement Windows 8, Sharepoint, les équipes agiles et la qualité logicielle. « De nouveaux modèles, des concepteurs et des outils de test et de débogage pour supporter la plateforme Windows 8, y compris

le support pour les applications Windows Store Windows 8 avec un simulateur. Analyse de code pour Windows Phone 8 applications. Cette mise à jour permet également de cibler Windows XP avec des applications en C++ natif », indique Microsoft.



Retour sur les Epitech Innovative Project 2012



Les 16 et 17 novembre derniers, les étudiants de l'Epitech présentaient leurs projets d'étude : les Epitech Innovative Project. Parmi les dizaines de projets présentés, deux nous ont particulièrement impressionnés par leur conception, les idées techniques et bien entendu leur réalisation concrète. Il est toujours difficile de faire un choix, car de nombreux autres projets auraient mérité d'avoir leur place ici.

François **Tonic**

► RealCar : piloter une voiture

Notre génération a vécu l'avènement du jeu vidéo, ces derniers possédant des moteurs physiques et des graphismes toujours plus bluffants, de plus en plus coûteux et complexes à réaliser. Nous atteignons les limites du réalisme dans les jeux vidéo "classiques".

Deux constats : Internet est partout, la multiplication des objets connectés. RealCar s'inscrit dans cette tendance. L'idée est de piloter à distance, par vidéo interposée, des véhicules miniatures bien réels, dans des décors ludiques immersifs.

> Cadre du projet

Dans le contexte du développement des EIP, nous avons le désir d'aller au-delà de la

simulation vidéo-ludique traditionnelle (créneau dans lequel tous les éditeurs s'engouffrent). Notre équipe, composée de 9 personnes aux compétences complémentaires, a donc choisi d'exploiter directement un objet physique évoluant dans un environnement réel. Voyons cela plutôt comme un nouvel objet ludique pilotable via Internet. Nous possédons l'équivalent d'un moteur graphique et physique qui vaut des milliers d'heures de programmation et d'algorithmie avancée : ainsi est né RealCar.

> Objectifs

Au départ, les objectifs concrets d'exploitation n'étaient pas bien définis. Tout comme la société Parrot lorsqu'elle développait ses premières versions de l'AR Drone, notre réflexion nous a orientée vers un premier produit accessible et apprécié de tous : les courses de voitures. Le choix de l'architecture et des technologies a été mûrement réfléchi. Fort d'une expérience antérieure sur du matériel embarqué à base d'IGEP, nous avons opté pour une infrastructure 3 tiers reposant sur du matériel de la société Gum-

stix, un serveur maître sous Debian et un client Windows/Linux en SDL/Qt. La communication entre le serveur maître et les voitures repose sur le WIFI, technologie la plus adaptée pour le moment, offrant une stabilité et des débits satisfaisants.

> Développement

Tous les développements ont été réalisés en Agile, répartis par équipe de 2 à 3 personnes. La jonction des briques se faisait régulièrement bien que les équipes étaient disséminées à travers le monde pour leur 4e année de formation.

> Serveur de jeu

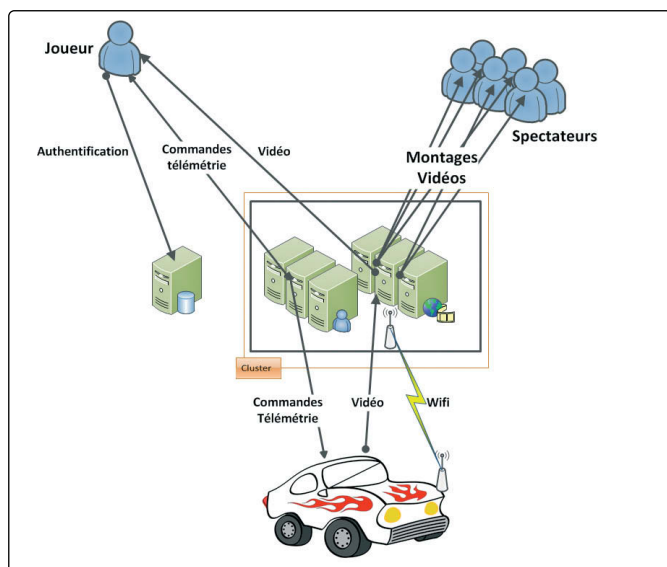
Un serveur maître (du jeu) contrôle l'ensemble du système, il a été réalisé en C++/Boost et coordonne l'assignation des commandes entre joueurs et voitures et les flux vidéo en retour.

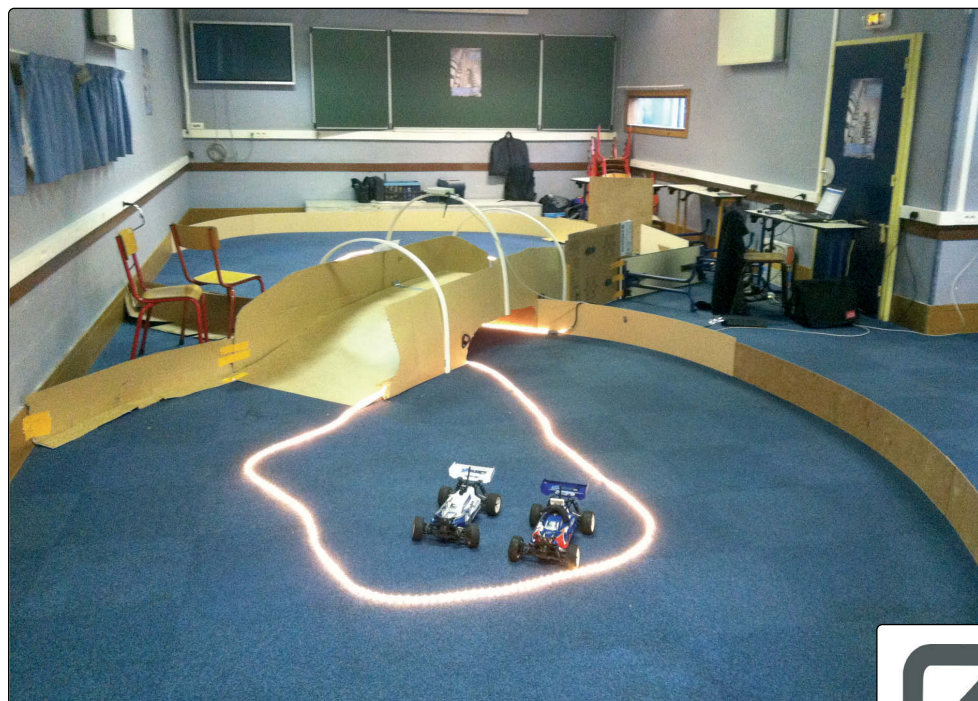
Le serveur supporte plus d'un millier d'objets connectés simultanément. Les objets connectés communiquent avec ce serveur, ainsi que les clients de jeu.

Les commandes émises par les clients de jeu aux objets connectés utilisent simultanément du TCP et de l'UDP pour assurer une réactivité instantanée.

> Objet connecté : Véhicule visio-piloté

Nous avons exploré plusieurs possibilités de châssis physique tels qu'un Crawler et un tank avant de nous fixer sur un modèle plus simple et plus fun que nous avons tous piloté dans notre jeunesse : un Buggy. Nos prototypes sont équipés d'une Gumstix Overo





COM intégrant un ARM Cortex-A8, ainsi qu'une carte additionnelle Pinto-TH et une extension maison pour l'alimentation et les servomoteurs.

L'électronique des servomoteurs est directement pilotée par les PWM – sortie analogique générant un signal logique compréhensible par les servomoteurs – connectés au processeur ARM embarqué. Quand la voiture boote, elle se connecte au wifi RealCar, démarre un daemon qui l'enregistre sur le master.

Ce daemon pilote la camera et envoie le flux vidéo au serveur en RTP. Des mini-caméras vidéos 480p fixées sur la voiture et munies d'objectif grand angle nous assurent une captation vidéo de qualité. Ainsi nous sommes capables de diffuser ce flux aux clients de jeu à plus de 60 fps.

> Proxy vidéo

Un proxy vidéo permet la diffusion publique d'un ou plusieurs véhicules en course, le stream se fait sur une voiture prise au hasard.

> Client de jeu

Le client de jeu permet aux joueurs de commander la voiture et de recevoir la vidéo de la course en temps réel. Cette partie logicielle a été réalisée en C++/SDL et repose sur GStreamer pour la réception du flux vidéo.

Cette dernière partie a été la plus sensible et nous avons exploré plusieurs voies avant de trouver la solution la plus adaptée.

D'autres solutions sont toujours à l'étude et nous sommes encore en attente de validation.

Le lag étant le principal obstacle à la fluidité requise par ce type de jeu, nous sommes cependant parvenus à le réduire suffisamment pour que celui-ci n'affecte pas le pilotage des voitures. Actuellement, la vidéo est transmise en RTP avec GStreamer et est encodée côté client (voiture) grâce au codec vidéo H264. La vidéo est envoyée au client final en UNICAST. Les commandes de jeu sont gérées sur clavier ou via un pad de Xbox. Il est également prévu de développer les retours haptiques pour le retour de force. Notre client est multiplateforme, il tourne sous Windows et sous Linux.

> Interface tablette tactile

Pour opérer en condition réelles, nous avons dû développer une application Android 4.0 qui est une véritable interface d'administration du jeu. Un opérateur sur le terrain dispose d'une tablette qui lui donne accès à toutes les fonctions du jeu.

Elle n'est pas indispensable mais elle permet de gérer les courses facilement, directement dans le feu de l'action. C'est elle qui gère les interactions entre le parc de voitures et le pool de joueurs.

Grâce à cette application, nous pouvons visualiser en temps réel les joueurs qui se connectent à nos serveurs, puis nous pou-

vons lui assigner une voiture disponible et enfin lancer la partie une fois que tous les joueurs sont prêts à entrer dans l'arène. Nous avons défini une API d'administration souple et complète pour nos besoins internes. La tablette utilise donc cette API pour présenter une interface adaptée. Notre application permet par exemple d'envoyer une commande de déconnexion au serveur qui transmettra l'information à la voiture et au client.

Le joueur est informé de sa déconnexion et la voiture redevient disponible dans le parc, tout ceci en temps réel.

La tablette permet également d'avoir le retour des différents capteurs de la voiture, comme l'état du véhicule, la batterie ainsi que d'autres informations télémétriques.



> Produit final et situations réelles

Après plus d'un an de recherche et développement, nous avons enfin eu l'occasion d'exploiter RealCar en conditions réelles.

Au Forum {EIP Epitech}, nous avons mis en place un circuit en huit d'une vingtaine de mètres de long, avec des obstacles, une mise

en condition (lumières, tunnel, pont) et deux opposants sur la piste. Les modèles réduits de voitures étaient bridés à 20% de leurs capacités soit 8 - 10 Km/h.

Des bêta-gamer se sont alors affrontés sur le terrain, en vue embarquée dans nos prototypes de voitures RC.

> Conclusion

L'intérêt croissant et la curiosité manifestée pour ce produit a été pour nous une énorme source de motivation, l'accueil réservé par le public étant la meilleure des récompenses. Nous avons recueilli de nombreuses remarques et suggestions et nous allons maintenant augmenter la réalité avec les nombreux éléments d'un jeu traditionnel. Les axes d'utilisation du produit et les dérivés sont très divers.

Nous continuons actuellement nos recherches, l'amélioration constante des technologies que nous utilisons nous aide dans ce sens. L'avènement de la 4G ouvre aussi la voie à d'autres opportunités. Une révolution est en marche.

Jean-Christophe **Baudier**, Archibald **Picq**, Romain **Poirson** et Steven **Martins**, Étudiants en 5e année à Epitech.

► C++ML : programmation visuelle en C++

C++ Modeling Language est tout d'abord un langage de conception logicielle et de programmation visuelle ; enfant spirituel du diagramme de classes de l'UML, il implémente l'intégralité des concepts propres au C++. Mais le terme C++ML désigne aussi un éditeur graphique, permettant de modéliser aisément ses conceptions.

Ce projet a vu le jour à Epitech, au cours de la scolarité de 7 grands amateurs de C++, qui eurent l'idée de créer un langage de modélisation propre au C++. Dans le cadre des *Epitech Innovative Project*, ils se sont appliqués à concevoir un langage graphique, évoluant parallèlement aux standards C++ ; le dernier en date étant le C++11. Désormais, la version "draft" de C++ML est entièrement rédigée, et l'éditeur est disponible en version bêta.

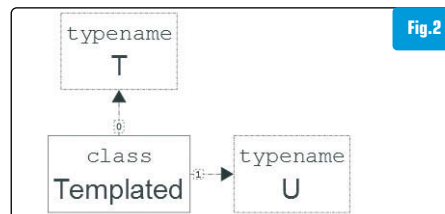
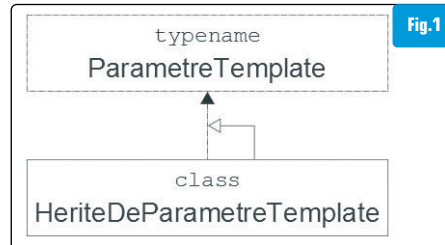
Contexte

De nombreux développeurs utilisent l'UML pour les aider lors de la conception de leurs projets, ou pour documenter du code déjà écrit. Bien que très puissant, l'UML est prévu pour de la programmation orientée objet, mais pas pour représenter les autres paradigmes présents en C++, notamment toute la surcouche ajoutée par les templates. En effet, représenter une classe recevant un paramètre template est simple dans un diagramme de classes UML, mais il est impossible de définir des liaisons avec ce même paramètre template. De nombreux autres cas de figures posent de vrais problèmes en UML. Il devient alors difficile de présenter des conceptions C++ complexes dans un diagramme de classes UML, sans y ajouter des annotations. Et malheureusement, dans ce cas-là, le développeur passe à côté de toute génération de code.

Le snippet de code numéro 1 donne un exemple de code C++ somme toute assez simple, mais néanmoins très ardu à représenter en UML. La [Fig.1] représente ce code en C++ML.

Le langage

C++ML est basé sur le diagramme de classe UML, la plupart des éléments présents en UML sont réutilisables en C++ML. Cependant, les méta-informations apportées par les templates sont représentées par l'ajout d'un nouveau type de classificateurs et d'un nouveau type de liaisons : les méta-classificateurs et les méta-liaisons. Ces derniers sont



dessinés avec des traits pointillés entrecoupés de points (trait, point, trait point, ...). Un ensemble de modifications mineures a aussi été appliqué à l'UML, afin d'avoir une cohérence entre C++ML et la syntaxe du C++.

Les représentations de méta-informations sont les suivantes :

- 1 méta-classificateur
- 5 méta-liaisons
 - la méta-liaison "[a comme] paramètre template"
 - la méta-liaison "(est une) spécialisation de template [de la classe]"
 - la méta-liaison "[a comme] paramètre spécialisant"
 - la méta-liaison "composition de type"
 - la méta-liaison "friendship"

Le snippet 2 et la [Fig.2] représentent une classe "Templated" templétée sur 2 paramètres T et U.

Les autres changements concernent notamment l'ajout d'un type au sommet de chaque classificateur ; afin de pouvoir faire la différence entre *class*, *struct* et *union* par exemple. La représentation de fonctions globales est maintenant possible via l'écriture d'un classificateur avec comme type celui de la fonction. La [Fig.3] et le snippet 3 représen-

tent la fonction globale "TerraformedPlanet Terraform(Planet);".

On peut aussi spécifier la visibilité d'une liaison d'héritage, via l'adjonction en bas de la hampe de la liaison d'héritage d'un rectangle en pointillé contenant directement la visibilité en question {*public*, *protected* ou *private*} ou respectivement {+, # ou -}.

Les fonctions et données sont maintenant spécifiées sous leur syntaxe C++, dans une question de cohérence avec ce dernier.

C++11 fait partie intégrante de C++ML, avec le support, entre autres, des templates variadiques ou des énumérations fortement typées, gérées quant à elles, sous la forme d'un enum agrégé d'une classe. Le support de certaines fonctionnalités faisant partie des versions C++ à venir (les concepts par exemple) est déjà implémenté.

> L'éditeur

Un éditeur écrit en C++, est déjà disponible en bêta. Le but est qu'il soit accessible librement à tous, et c'est pour cela qu'il sera distribué sous une licence très permissive : *new BSD*. La portabilité est un point important, pour qu'un maximum de développeurs ait accès au logiciel, il doit pouvoir tourner sous le plus grand panel d'architectures possible. L'interface graphique est abstraite pour permettre à l'utilisateur d'utiliser autre chose que l'implémentation *Qt* fournie par défaut, qui pourrait ne pas lui convenir. Cet éditeur permettra la génération de code C++ à partir d'un diagramme, et inversement, pourra passer d'un code C++ à un diagramme. Il sera interfaçable avec la plupart des IDE et logiciels existants, afin de ne pas perturber les habitudes de travail des développeurs.

Site officiel du projet : <http://cplusplusml.org>.

Remerciement à M. François Carrubba.

Nicolas Albeza, Matthieu Cholley,
Romain Gay, Laurent Humez,
Pierre-Yves Le Borgne, Jérémy Lecerc
et Tony Tanon

Étudiants en 5^e année à Epitech

communautés

Lancement réussi d'Agile.Net France

Une nouvelle communauté agile / ALM vient d'être lancée. Le 3 décembre dernier, Agile .Net France a officiellement tenu sa première réunion. Cette association à but non lucratif dont l'objet est la promotion des méthodologies agiles dans les environnements Microsoft en France, a été créée début Septembre par Arnaud Hego et Luc Legardeur, respectivement Président et Directeur Général de Cellenza (Luc étant également le fondateur du French Scrum User Group) et a reçu le soutien d'un parrain de prestige en la personne de **Jeff Sutherland**, le père fondateur de la métho-



de Scrum, lui aussi présent lors de la soirée. « Cette initiative est née du constat que de nombreuses initiatives associatives et/ou communautaires existaient en France autour de l'agilité et autour des technologies Microsoft, mais qu'aucune ne réussissait le mariage des deux jusque-là ! Pourtant il y avait une attente réelle... », nous explique Arnaud Hego.

Pour en savoir plus : www.agiledotnetfrance.org

GDG Nantes : c'est quoi une communauté développeur Google ?

Les Google Developers Groups (GDG) sont des groupes d'utilisateurs intéressés par les technologies de développement Google; Tout, depuis les plateformes Android et App Engine, jusqu'au API des produits Google comme YouTube et le Google Calendar, et également jusqu'aux initiatives comme OpenSocial. Un GDG peut prendre de nombreuses formes... Depuis quelques personnes se rassemblant pour regarder les



dernières vidéos de Google, à de larges conférences avec des démonstrations et des présentations, ou encore des événements de codage tels les hackathons. Le GDG de Nantes est né de l'envie de présenter, d'expliquer et de fédérer les développeurs autour des outils et technologies Google dans la région nantaise. L'activité principale du GDG Nantes consiste à organiser des événements (conférences, forums, tables rondes, ateliers...) autour des technologies Google. Il est prévu d'organiser des événements en soirée (à partir de 18h) proposant un sujet de conférence d'une durée de 90 minutes. Ces soirées sont clôturées par un buffet qui permet aux participants de se rencontrer et de discuter librement. Le GBG souhaite aussi organiser des bar-camp. Pour en savoir plus : <http://gdgnantes.com>

Février 2013

Fosdem (Bruxelles)

En quelques années, le FOSDEM est devenu l'événement open source et du logiciel libre incontournable en Europe. Toujours organisé à Bruxelles, la conférence rassemble plus de 5000 personnes, 400 sessions. A l'heure où nous bouclons le magazine, le programme n'était pas encore disponible.

Site : <http://fosdem.org>

6000 hackers
400 lectures
2 days
free entrance

developers hackers open source non-commercial lightning talks main tracks devrooms conference

Free and Open Source Software Developer's European Meeting

FOSDEM13

2 - 3 Feb 2013 - Brussels, BE
<http://fosdem.org>

Get Ready to meet in Amsterdam

5-6 February, 2013 | Amsterdam, The Netherlands
Amsterdam RAI Convention Centre

BlackBerry Jam 2013 (Amsterdam)

RIM organise une nouvelle tournée mondiale pour les développeurs : BlackBerry Jam. En Europe, elle aura lieu du 5 au 6 février 2013 à Amsterdam, quelques jours seulement après la présentation officielle de BlackBerry 10. La conférence a pour objectif d'expliquer, présenter l'ensemble de la nouvelle plateforme, les premiers terminaux. De nombreuses sessions techniques seront jouées. Vous pourrez rencontrer les ingénieurs RIM et échanger avec d'autres développeurs BlackBerry. Site : <http://www.blackberryjamconference.com/europe>

Microsoft TechDays 2013 (Paris)

Du 12 au 14 février, au Palais des Congrès de Paris, la grand'messe des technologies Microsoft se déroulera sous le signe de Windows 8, Windows Phone 8, Windows Server 2012, Windows Azure, etc. Comme chaque année, les 3 sessions plénières seront thématiques : développeur, IT Pro et innovation technologique. Plus de 300 sessions, sur plusieurs dizaines de thèmes, seront disponibles ainsi que de très nombreux partenaires dans la partie exposition. Vous pouvez dès maintenant vous inscrire (gratuitement) et organiser votre agenda ! Site officiel : <http://www.microsoft.com/france/mstechdays/>

agenda

JANVIER

Le 24 janvier 2013, CNIT Paris La Défense, **Salon Lesjeudis.com**. Plus d'une trentaine de recruteurs seront présents. www.lesjeudis.com

FÉVRIER

Le 12 février 2013, CNIT Paris La Défense, **NIDays 2013**, dédié cette année, à la conception, au test et au contrôle : <http://france.ni.com/nidays>

ETRANGER

Du 23 au 25 janvier 2013, Rome, **40e colloque sur les principes des langages de programmation**. Forum de discussion sur tous les aspects des langages et des systèmes de programmation. <http://popl.mpi-sws.org/2013/>

ATELIER DE GÉNIE LOGICIEL PROFESSIONNEL

DÉVELOPPEZ 10 FOIS PLUS VITE

N°1 EN FRANCE

WINDEV®



- Windows 8 (7, XP,...)
- Linux
- Mac
- Internet
- Intranet
- Windows Mobile & CE
- Android
- iPhone & iPad

- **WINDEV** est **LE** logiciel qui permet de créer les logiciels.

Cette plateforme professionnelle permet de créer tous les types de logiciels, pour tous les domaines, pour tous les types de matériels, dans tous les environnements, pour toutes les volumétries.

Votre code, vos fenêtres, vos données, vos rapports,... sont **compatibles**.

Vous aussi, développez 10 fois plus vite pour **toutes les plateformes**.

Fournisseur Officiel de la Préparation Olympique

Elu «Langage le plus productif du marché»

VERSION EXPRESS GRATUITE
Téléchargez-la !



Je te veux

► **DEMANDEZ VOTRE DOSSIER GRATUIT**

Dossier gratuit 260 pages sur simple demande. Tél: 04.67.032.032



918
NOUVEAUTÉS

www.pcsoft.fr

OFFRE ANNIVERSAIRE

6 MOIS

GRATUITS



DOMAINES | EMAIL | HÉBERGEMENT | E-COMMERCE | SERVEURS

*Offres anniversaire valables sur les packs hébergement, une sélection de serveurs dédiés et sur la configuration de base du Serveur Cloud Dynamique. Offres soumises à des frais de mise en service et à un engagement de 12 mois. Offres sans durée minimum d'engagement également disponibles. **Offre domaines** : les noms de domaine sont à partir de 0,99 € HT/an (1,18 € TTC/an) la 1^{re} année, puis le prix habituel s'applique. Conditions détaillées sur 1and1.fr.

1&1 HÉBERGEMENT WEB

Cette année, 1&1 fête ses 25 ans ! Depuis notre création, nous avons sans cesse développé notre savoir-faire et aujourd'hui, nous comptons parmi les leaders mondiaux de l'hébergement Web avec plus de 11 millions de contrats clients, 5000 employés et 5 data centers haute performance.

Nos 25 ans d'expérience nous permettent de vous faire bénéficier au quotidien de solutions Web haute fiabilité et d'un service expert pour garantir l'avenir de vos projets sur la Toile. A l'occasion de notre anniversaire, nous vous offrons jusqu'à 6 mois gratuits sur une sélection de produits !

1&1 PACKS HÉBERGEMENT



- Sécurité maximale : hébergement en parallèle dans 2 data centers 1&1
- Protection complète de votre site (1&1 SiteLock) et de vos données avec la fonction Restauration de l'espace Web
- NOUVEAU ! PHP 5.4 inclus
- Accès à 65 applications Click & Build : WordPress, Joomla!, Drupal...
- Choix des O.S : Windows ou Linux
- Trafic illimité

6 MOIS GRATUITS

puis à partir de 1,99 € HT/mois
(2,38 € TTC/mois)*

1&1 SERVEUR CLOUD DYNAMIQUE



- Maîtrise des coûts : configuration et facturation à l'heure
- Opérations sécurisées sur vos VM : 1&1 Snapshot
- Citrix Xen Server sur architecture AMD™ multi-cœurs (AMD™ Opteron 6272)
- Appli mobile de gestion de vos VM
- Accès root via SSH
- Trafic illimité, bande passante de 100 Mbps

3 MOIS GRATUITS

puis à partir de 24,99 € HT/mois
(29,89 € TTC/mois)*

1&1 SERVEURS DÉDIÉS



- Processeurs dernière génération : Intel® et AMD™
- Appli gratuite : contrôlez votre serveur depuis votre mobile
- Parallels® Plesk Panel 11
- Disponibilité réseau de près de 100%
- Choix des O.S : Linux (CentOS, openSUSE, Debian, Ubuntu) ou Windows (en option)
- Trafic illimité, bande passante de 100 Mbps

3 MOIS GRATUITS

puis à partir de 99,99 € HT/mois
(119,59 € TTC/mois)*

Réservez votre nom de domaine à partir de 0,99 € HT/an*



0970 808 911
(appel non surtaxé)

1and1.fr

De Wargame à XAML : les multiples vies d'un geek-développeur

Samuel Blanchard est responsable du développement et de l'innovation chez Naviso, une société spécialisée dans les solutions de gestions innovantes et le développement, mais c'est avant tout un développeur passionné du Web et de l'embarqué. Modérateur de forums et rédacteur de blogs, Samuel s'intéresse principalement aux communautés Microsoft et plus précisément à Windows Phone et Windows 8. Il est MVP Windows Phone.

Comment es-tu tombé dans l'informatique et plus spécialement dans le développement ?

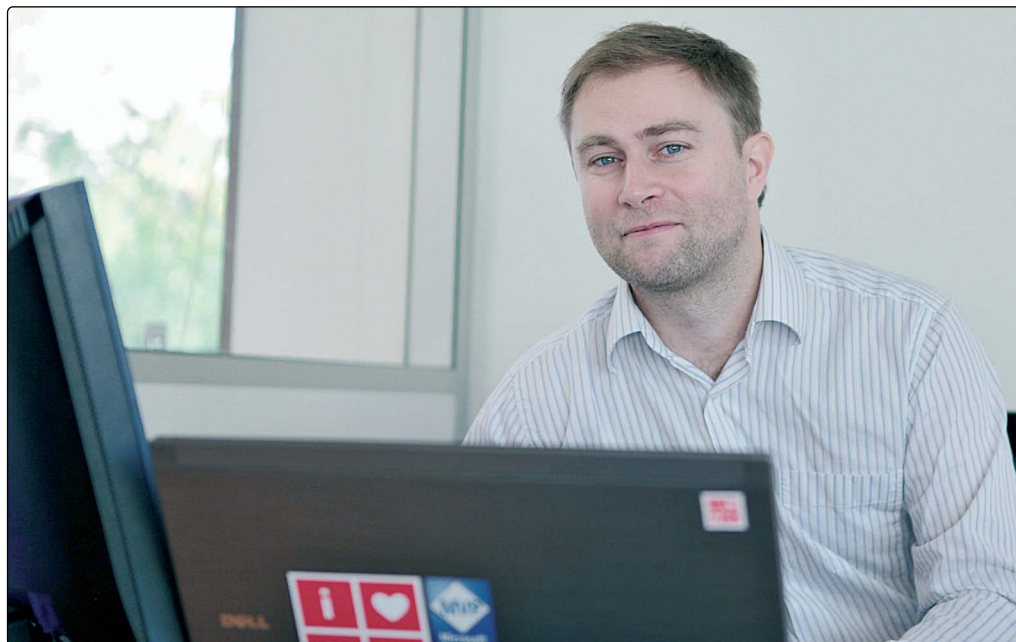
Je suis « tombé en amour » de l'informatique plusieurs fois dans ma vie. La première, lorsque je tapotais sur mon minitel en me prenant pour le héros du film Wargame. Atari et Amiga sont ensuite devenus mes jouets de prédilection.

La seconde fois, quand j'ai découvert que l'on pouvait développer des jeux pour Gameboy en assembleur et en C.

La troisième, lorsque j'ai appris le XAML. Je reste encore émerveillé par le potentiel de ce langage. Par exemple, sa capacité à changer complètement l'apparence de ses contrôles sans changer leur fonctionnalité première est un rêve pour celui qui développe des interfaces utilisateurs.

Bref, l'important est de préserver l'envie et cela ne peut se faire qu'en restant curieux.

Pour toi, qu'est-ce qui fait que l'on aime toujours et encore le développement, la technique ?



Le renouvellement. Lorsque l'on passe plusieurs années sur la même technologie, on devient souvent un expert mais l'on s'use à répondre aux mêmes questions, aux mêmes problématiques.

Il est donc important de changer de technologie régulièrement pour ne pas rentrer dans une certaine routine. C'est la beauté de ce métier.

L'échange est aussi un facteur important pour partager sa passion et ne pas rester prisonnier de son expertise.

Participer à des communautés techniques est un bon moyen de conserver la flamme. A la fois pour aider les développeurs junior mais aussi pour apprendre de nouvelles techniques, de nouvelles façons de voir.

Tu as gardé un regard très geek : gadget, veille techno. C'est important pour ton job et ta passion ?

Oui c'est important. Plus jeune je voulais développer des jeux vidéo sur consoles. Dès qu'ont émergés des kits de développement pour Gameboy je me suis mis au travail. J'ai créé un jeu, Crazy Zone, pour participer à un concours que j'ai remporté. Ce jeu s'est retrouvé par la suite intégré pendant quelque temps dans les FreeBox qui disposaient alors d'un émulateur Gameboy.

Mon bureau

Je travaille dans un bureau ouvert où je suis en relation constante avec mon équipe. J'ai eu plusieurs fois la possibilité de travailler dans un bureau fermé mais je pense que la proximité avec ses équipes est une chose importante aussi bien socialement que techniquement. La musique m'est indispensable pour pouvoir me concentrer sur un projet de développement. Je n'écoute pratiquement que des musiques de films. Mes écouteurs sont branchés en permanence sur mon ordinateur, ce qui me permet de pouvoir m'isoler dès que je le souhaite sans forcément changer de bureau. Mon bureau en lui-même est constitué d'un ensemble d'écrans et de machines dont la place varie selon la priorité du moment. Ces derniers mois, mon Windows Phone et ma tablette Win8 sont au centre de mes intérêts.

Aujourd'hui cette passion pour la petite console de Nintendo ne m'a pas quitté puisque je développe un émulateur Gameboy Color sur Windows Phone (Purple Cherry). Tout cela m'a appris des techniques d'optimisation que j'intègre dans mes réalisations professionnelles et que je communique à mes équipes et par le biais de mon blog.



La veille techno fait partie intégrante de mon travail et me permet de détecter les technologies qui pourraient intéresser Naviso. Elles se retrouveront ensuite dans les applications de nos clients ou seront abordées lors de conférences telles que les Techdays ou des BarCamps.

Etre développeur n'est pas toujours facile : pression, évolution constante, frustration des projets et des "chefs", c'est quoi pour toi être développeur aujourd'hui ? Le job a-t-il changé depuis tes débuts ?

Oui il a changé. Plutôt en bien d'ailleurs. La pénurie de développeurs en France fait que l'on ne nous considère plus comme des barbus échappés d'un garage.

C'est un poste de plus en plus important dans l'entreprise et bien que nous ne soyons pas au niveau des américains puisqu'il est toujours nécessaire de manager pour grimper dans la hiérarchie les mentalités évoluent rapidement. « Fiers d'être développeurs » est un leitmotiv qui me tient à cœur. La pression et l'adaptation aux nouvelles technologies sont inhérentes au poste, mais heureusement elles sont gérables tant que l'on reste organisé.

Et en dehors du boulot, qu'est-ce que tu aimes faire ? Comment trouves-tu l'équilibre entre travail, vie privée, passion, famille ?

J'aime le cinéma, courir, mais comme tout bon passionné qui se respecte je continue de développer des applications sur mon temps libre. Pour que cela puisse fonctionner au quotidien, il est important que la passion ne déborde pas sur les moments familiaux.

J'ai la chance de pouvoir l'exercer au sein de mon entreprise, Naviso, mais aussi, de la partager avec certains amis.

J'essaie, néanmoins, de ne pas m'enfermer dans cette passion et de conserver un lien avec des amis dont les centres d'intérêts

sont aux antipodes des miens. J'ai la chance d'avoir une compagne, une famille et des proches qui m'ouvrent à leurs propres centres d'intérêt. De la richesse de nos échanges naissent aussi, parfois, de nouvelles idées d'applications.

L'équilibre, même s'il est parfois difficile à trouver, demande, de part et d'autre, de la compréhension, mais aussi de la disponibilité et des moments de partage.

Montrer le fruit de mes réflexions et de mon travail à mes proches leur permet de mieux comprendre mon implication, parfois dévorante, dans mes projets. A ce jour, je crois pouvoir dire que j'ai trouvé l'équilibre que je recherchais.

Peux-tu nous présenter ton quotidien en quelques mots ?

Mon travail consiste tout d'abord à gérer une équipe de développement et les projets qui lui sont affiliés. Nous travaillons sur plusieurs portails extranets utilisés par des centaines de personnes au quotidien.

Nous restons concentrés sur le développement de nouveaux modules mais également sur l'amélioration et l'évolution de modules déjà mis en place. Cela demande de couvrir d'autres compétences que le métier de développeur.

Ainsi, il n'est pas rare que nous appelions les clients pour obtenir des informations complémentaires sur des problèmes qu'ils pourraient rencontrer, ou que nous devions installer nos applications et leurs composants sur nos serveurs. Je reste également en alerte sur les technologies en devenir et je mets en place des projets utilisant les plus prometteuses afin d'en déterminer le potentiel réel. Je prends aussi du temps dans la semaine pour gérer mon blog (<http://blog.naviso.fr>), forums, communautés, et autres pages Facebook. Je suis présent particulièrement sur des forums consacrés à Windows Phone.

Comment vois-tu ton job évoluer ?

Mon job a évolué dernièrement pour prendre en compte une dimension d'innovation qui me tient particulièrement à cœur.

A terme, le management des équipes s'effacera peut-être, au profit de l'orientation innovation et développement. Cette évolution est, je pense, liée au changement de mentalité concernant les développeurs et ce qu'ils peuvent apporter à l'entreprise.

Mon job va aussi évoluer par le biais des opportunités que mon entreprise, elle-même, crée pour l'équipe de développeurs. Les clients de Naviso, essentiellement des entreprises que nous équipons en logiciels de gestion, nous sollicitent chaque jour davantage pour réaliser des développements spécifiques, des applications sur mesure pour PC, tablette ou téléphone.

De fait, le pôle développement que je supervise prend de l'importance et devient un levier de croissance pour Naviso.

Des conseils aux étudiants et devs qui nous lisent ?

Depuis quelques années, les communautés de développeurs, chacune experte dans son domaine ou sa technologie, sont devenues incontournables. Grâce à elles, notre travail est devenu plus collaboratif, l'autoformation facile et perpétuelle.

La nouvelle génération de développeurs a ce réflexe communautaire et doit être consciente de la force que cela lui confère.

Apprendre des autres, mais aussi diffuser son savoir, son expertise sont des atouts dont peu de communautés peuvent se prévaloir. Pour progresser rapidement il faut avoir des objectifs définis. Même si ceux-ci semblent ambitieux, ne vous découragez pas. Certains de mes projets ont mis plusieurs années à se concrétiser. Il faut simplement ne pas avoir peur de les mettre de côté pour les reprendre plus tard avec une vision différente et plus mature. #

Toi aussi, parle de ton métier de développeur ! Contact : f tonic@programmez.com

L'information permanente

• L'actu de Programmez.com : le fil d'info quotidien

• La newsletter hebdo :

la synthèse des informations indispensables.

Abonnez-vous, c'est gratuit !

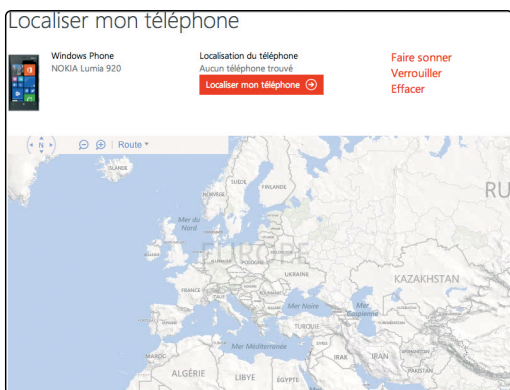
www.programmez.com



Sécuriser et protéger votre matériel informatique

Les vols d'ordinateurs (notamment portables), de smartphones explosent depuis quelques années. Il existe plusieurs solutions, souvent complémentaires, pour sécuriser et protéger votre matériel. Nous vous proposons quelques idées rapides à mettre en place.

Ces protections sont de différentes natures : physiques (câbles de sécurité, clé chiffrée, tatouages, accès biométrique, etc.) ou non (assurances, effacement à distance). Aucune solution n'est efficace à 100 % et parfois, cela peut se retourner contre vous. En août dernier, un journaliste informatique américain a connu une cruelle mésaventure : un hacker avait réussi à pénétrer ses comptes Gmail, iCloud, Amazon, suites à des failles dans les procédures d'authentification, notamment par téléphone...



> Le verrou antivol de type Kensington

Une des sécurités les plus basiques est le verrou antivol Kensington que l'on retrouve sur de nombreux ordinateurs et périphériques (notamment sur les NAS). Cet antivol se présente sous la forme d'une petite encoche rectangulaire permettant de fixer un verrou, un câble compatible. En cas de vol, ou de tentative, si le voleur arrache de force l'antivol, le boîtier sera endommagé et donc parfaitement visible. Bien entendu, cela n'empêchera jamais le vol mais cela peut dissuader et surtout ralentir le voleur. Les câbles Kensington sont assez résistants. Ne pas oublier d'enrouler le câble au bureau, chaise, etc. (si le câble n'est pas fixé à quelque chose, à quoi sert-il ?).

Avantages :

- standard
- diversité des câbles
- dissuasion visuelle

Inconvénients :

- prix parfois élevé des kits de sécurité
- pas présent sur tous les matériels (surtout sur les ultrabook, les tablettes)
- n'empêche pas le vol : il le ralentit seulement.

Assez étonnant, le constructeur Devolo équipe certaines gammes CPL d'une encoche Kensington !

> Tatouage et assurance

Le tatouage consiste à apposer (ou par technique de gravure) sur un matériel une plaque d'identification, très difficilement retirable pour un voleur. Via un logiciel installé sur la machine, la société gestionnaire du tatouage récolte alors l'ensemble des numéros de séries permettant un suivi du matériel en cas de vol ou de perte, avec localisation et transmission des informations à la police. Parfois, une assurance vol est incluse dans le contrat de tatouage. Côté assurance, il existe de nombreuses offres contre le vol et la perte. Par exemple, LCL propose « Assurance Tous



Portables ». Mais attention : vérifiez bien la franchise, les plafonds imposés (par exemple : limite de 400 € pour un smartphone).

> Utiliser les services de géolocalisation

Parfois, on peut être un peu étourdi et égarer son portable ou un smartphone. Certains éditeurs / constructeurs proposent, en standard, un service de géolocalisation des terminaux. Windows Phone 8 propose la fonction « mon téléphone » pour localiser, verrouiller, effacer son smartphone. Apple propose depuis plus de 18 mois la fonction localisation via le service iCloud. Il permet de localiser, verrouiller et effacer un iPhone, iPad, un Mac. Cependant, attention. Assurez-vous que votre mot de passe de localisation soit assez compliqué et fort pour éviter tout piratage de votre service. D'autre part, quand cela est autorisé par le système, interdisez toute modification de configuration de localisation sur le terminal, sans taper un mot de passe. Car après, rien de plus de simple que d'effacer à distance le contenu de votre machine ou téléphone !

François Tonic

Pour aller plus...

Pour sécuriser un peu plus votre machine, vous pouvez opter pour une authentification biométrique. Il existe quelques claviers intégrant une lecture d'empreinte digitale comme chez Lenovo ou Cherry, mais cela reste encore rare et pas toujours très bon marché. Des lecteurs indépendants peuvent être utilisés (ex. : Eikon). Quelques constructeurs intègrent par défaut un tel lecteur sur les portables (Acer, Lenovo, HP) mais essentiellement sur les gammes professionnelles. Vous pouvez aussi utiliser des clés USB, ou des disques cryptés pour augmenter la sécurité de vos données. Automatiquement, toutes les données stockées sur la clé seront chiffrées (256 bits le plus souvent). Mais attention : assurez-vous de la fiabilité de la clé (la corruption de données peut rapidement rendre inaccessible vos fichiers). Vous pouvez aussi utiliser des disques ou des SSD cryptés par défaut. Bémol : le prix souvent élevé de ces solutions.



Meetup SkySQL à Paris : Cloud, MySQL, MariaDB

La communauté des utilisateurs professionnels de MySQL s'est retrouvée à Paris, le 13 décembre dernier, dans une ambiance tropicale pour envisager le futur de MySQL, à l'invitation de SkySQL. L'objectif était d'aborder les évolutions de MySQL, MariaDB, et le Cloud.

Le message de SkySQL, qui repose sur son expérience terrain chez les clients, permet une adoption plus rapide de MariaDB. Serge Frezefond, ingénieur consultant à SkySQL a présenté ainsi au meet-up les avantages d'un passage rapide à MariaDB : « Si vous avez des soucis de performances avec MySQL 5.5, essayez MariaDB 5.5 vous serez surpris ! » Rappelons que MariaDB 5.5 est totalement compatible avec MySQL 5.5, notamment car il est écrit par la core team originelle, avec des nouveautés et des optimisations de performances laissant loin derrière le MySQL d'Oracle.

> Fondation MariaDB, l'avenir open-source de MySQL

Après les déboires de rachats multiples du serveur de bases de données le plus utilisé au monde, MySQL a fini entre les mains d'Oracle. Ce dernier multiplie les séparations fortes entre le core Open-Source du serveur et de nouvelles extensions propriétaires. Le changement de licence par rapport à Sun a emprisonné les entreprises clientes en obligeant la désinstallation de MySQL si le contrat de support n'est plus contracté. Autant de raisons et de craintes, qui comme pour OpenOffice ont poussé les développeurs et leaders de ces projets à faire un fork. A l'instar de LibreOffice, digne successeur d'OpenOffice aujourd'hui structuré dans « The Document Foundation », ou encore de la plus connue des fondations, celle de Mozilla (qui édite le navigateur Firefox, le client mail Thunderbird et le prochain Firefox OS), MariaDB le fork de MySQL s'organise aussi en fondation, par la volonté de son créateur Monty Widenius, afin de lui garantir un développement pérenne. Cette annonce est un signe fort à destination des utilisateurs de MySQL sur l'avenir de leurs bases de données. Pour une grande majorité des entreprises, ce SGBD est devenu stratégique pour leurs développements et il est inconcevable de tomber dans le giron

d'une entreprise qui ne permette pas de garantir le bon fonctionnement et les évolutions futures. Notamment, car la base de données a été choisie pour son modèle Open Source.

Les trois acteurs à l'initiative de la fondation sont : Monty Program Ab qui regroupe une très grande partie des développeurs MySQL démissionnaires suite au rachat par Oracle, SkySQL qui regroupe les consultants et experts à l'époque de MySQL Ab, et Percona dont les équipes sont à l'origine du moteur de stockage InnoDB racheté par Oracle et qui ont fait leur propre fork de MySQL. Ces trois acteurs vont faire les bases de la fondation en attendant la constitution d'un premier bureau. La fondation se voit soutenue financièrement (actuellement 1 million d'euros) par plusieurs sponsors, principalement des entreprises clientes de MySQL, souhaitant un modèle de développement libre pour leurs bases de données.



MariaDB facilite le travail des sysadmins

SkySQL a présenté de son côté SkySQL Enterprise Data Suite regroupant différents outils d'aide aux utilisateurs et aux administrateurs comme la supervision avancée, l'anticipation de problèmes, la synchronisation automatique des bases et la resynchronisation en cas d'incident. On notera aussi l'arrivée de SkySQL Cloud Data Suite qui permet un déploiement automatisé d'une architecture MariaDB sur Amazon EC2.



Si toutes les conditions sont réunies pour faire de MariaDB le nouveau MySQL, il faut néanmoins composer avec une différence de planning liée aux utilisateurs de MySQL. En effet si les dernières versions des systèmes d'exploitation ont remplacé OpenOffice par LibreOffice, les utilisateurs de MySQL ont un rythme de changement de version plus lent qui se compte en années. Il est pratiquement acquis que les prochaines versions des OS intégreront MariaDB 5.5 en remplacement de MySQL, mais il faudra sans doute plusieurs années pour voir les entreprises mettre à jour leurs serveurs de bases de données. Actuellement, la grande majorité utilise encore la version 5.1, la v5.5 étant réservée aux nouveaux projets ou aux entreprises désireuses de bénéficier des dernières avancées.

Rappelons que MariaDB 5.5 est totalement compatible avec MySQL 5.5, tout en apportant des nouveautés et des optimisations de performances laissant loin derrière MySQL. Pour ceux qui souhaitent prévoir leur migration ou utiliser MariaDB pour de nouveaux projets, sachez que les fichiers de configuration et les binaires / utilitaires portent les mêmes noms que sur MySQL par Oracle. Cela n'aura donc aucun impact sur vos outils de déploiements / scripts.

Ressources :

<http://mariadb.org/> MariaDB
<https://kb.askmonty.org/en/> KnowledgeBase MariaDB
<http://www.skysql.com/> SkySQL
<http://config.skysql.com/> Cloud Data Suite
<http://www.skysql.com/enterprise> Enterprise Data Suite

Pierre-Henry Muller

Fondateur et dirigeant de DigDeo (www.digdeo.fr) entreprise d'infogérance / hébergement et d'expertise sur les logiciels Open-Source. Dans le cadre de l'optimisation des serveurs web pour la haute performance, DigDeo a remplacé avec succès Oracle MySQL par MariaDB dans le cadre de l'accompagnement proposé à ses clients.

Neo4j : l'autre univers du stockage

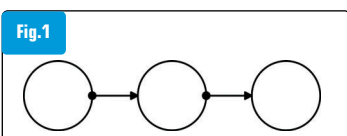
L'univers du stockage de données sur le web a beaucoup évolué depuis une dizaine d'années. Des fichiers textes préhistoriques, le stockage a rapidement évolué vers les bases de données relationnelles de type MySQL ou PostgreSQL. Les années 2000 ont cependant bouleversé cette hégémonie en propulsant le concept NoSQL sur le devant de la scène.

L'apparition de produits tels que CoucheDb, Redis, MongoDB, Memcache fut une véritable révolution dans l'univers du développement web : le « relationnel » n'était plus seul et pouvait intelligemment être secondé par d'autres technologies plus optimales pour certains usages (Facebook est un exemple typique d'un géant du web dont l'infrastructure repose autant sur le SQL que sur le NoSQL). Un choix pertinent d'une infrastructure de stockage fait désormais partie intégrante des prérogatives d'un architecte logiciel.

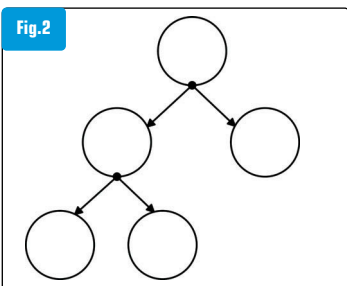
> Graphe

Un graphe est un objet informatique appartenant à la famille des structures de données. Ces dernières constituent les objets élémentaires manipulés par les algorithmes ; en choisissant une structure de données adaptée, le développeur aura d'autant de plus chances d'apporter une réponse optimale au problème à résoudre. Etre en mesure de reconnaître une structure de données est par conséquent essentiel pour un développeur.

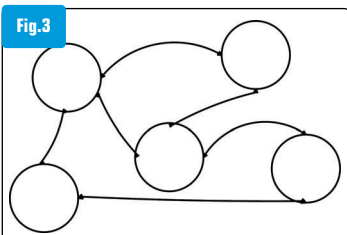
Certaines structures sont plus simples à appréhender que d'autres.



Dans le cadre d'une structure de type « liste », tout élément ne peut être relié qu'à un seul autre élément [Fig.1].



La structure d'« arbre » est plus complexe ; chaque élément peut avoir plusieurs descendants. On parle alors de « fils » reliés par des « arrêtes » à un « père ». Cette structure est notamment utilisée dans le domaine de la compression de données et de la compilation [Fig.2].



Toute complexe qu'elle soit, la structure d'arbre reste cependant triviale face au plus redoutable des objets informatiques : le graphe. Dans le cadre d'un graphe, chaque élément est relié à un ou plusieurs autres éléments. On parle alors de « nœuds » reliés par des « arcs » [Fig.3].

D'apparence simple, ces graphes se révèlent être des objets terriblement coriaces auxquels sont liés certains problèmes mathématiques parmi les plus complexes.

> Usage

Sans que nous en ayons toujours conscience, les graphes font partie intégrante de notre vie quotidienne. Un plan de métro peut être assimilé à un graphe, chaque station étant reliée à une ou plusieurs autres. Notre vie « digitale » profite également de ce concept,

notamment dans le cadre des réseaux sociaux : chaque membre dispose d'amis qui sont eux-mêmes en relation avec d'autres d'amis. Plus globalement, le réseau internet est un gigantesque graphe où chaque appareil est relié à d'autres par des connexions (câbles, fibres).

> Développement

Les sites web offrant toujours plus de fonctionnalités innovantes, il n'est pas rare qu'un développeur, en cherchant à structurer ses données, se retrouve à devoir manipuler un graphe. Un développeur « non initié » aura tendance à envisager une implémentation reposant sur les bases de données et le « SQL ». L'utilisation d'une table « nœud d'origine » « nœud de destination » alliée à des procédures stockées peut donner l'illusion d'une solution adaptée.

Aussi intelligentes soient-elles, ces implémentations ne parviendront jamais à assurer une scalabilité suffisante, à partir du moment où le graphe comportera plusieurs milliers de nœuds. Les algorithmes mis en œuvre pour manipuler les graphes sont spécifiques et nécessitent des connaissances expertes dans des domaines très pointus des mathématiques.

Compte-tenu de ce besoin, Neo a créé une solution spécifique pour venir en aide aux développeurs.

> Neo4j

Le serveur Neo4j permet de stocker vos données sous la forme d'un graphe. En plus de sa rapidité, de sa richesse fonctionnelle et de sa stabilité, Neo4j a l'avantage d'être particulièrement scalable : gérer des milliards de nœuds et d'arcs ne lui fait absolument pas peur. Autre aspect important, Neo4j gère de façon native la haute disponibilité et la redondance. Ces fonctionnalités se révèlent particulière-



ment séduisantes pour les sociétés dont le cœur d'activité repose sur un graphe.

A ce titre, Viadeo, réseau professionnel français, a sélectionné Neo4j pour stocker les relations entre ses membres.

Open-source et gratuit, le serveur Neo4j est disponible sur

neo4j.org. Ecrit en Java, il est disponible sur les principaux OS tels que Windows, Linux et OS X. Cette dernière plateforme sera notre plateforme de référence dans le reste de l'article.

Soutenu par une société commerciale (vendant du support et du conseil), le projet Neo4j profite à la fois de développeurs dédiés et d'une communauté très active sur les forums et listes de diffusion.

Autre élément non négligeable pour un développeur, la documentation de Neo4j disponible en ligne (<http://docs.neo4j.org/pdf/neo4j-manual-1.7.pdf>) se révèle particulièrement à jour et complète.

> Installation

Neo4j est distribué sous forme d'archive. Une fois l'arborescence extraite (ex. `tar xzvf neo4j-community-1.7.2-unix.tar.gz`), le serveur est démarré en se plaçant dans le répertoire `bin` et en exécutant la commande `neo4j start`. L'autre répertoire important est `data` ; il contient les données du graphe (`data/graph.db` et `data/keystore`) ainsi que les journaux (`data/log`) [Fig.4].

Le serveur « écoute » sur le port 7474. Il est possible de s'assurer que le serveur est bien démarré en accédant à l'url <http://localhost:7474>. Ce site permet de superviser l'état du graphe et offre différents outils présentés dans la suite de l'article [Fig.5].

> Communication

Communiquer avec le serveur Neo4j peut être réalisé par différents biais. La manière la plus « naturelle » consiste à exploiter les librairies Java livrées avec le serveur (dans le répertoire `lib`). Les développeurs ne souhaitant pas utiliser Java peuvent accéder à un large panel de fonctionnalités de l'API par l'intermédiaire de web services. Il devient alors possible de transmettre ses requêtes au serveur avec son langage de prédilection (en concevant une simple passerelle HTTP). Les web services fonctionnent en mode REST et utilisent le format JSON pour les échanges. Tous les appels web services sont basés sur l'url suivant : <http://localhost:7474/db/data>.

Par ailleurs, un accès à cette url racine retourne différentes informations sur le serveur (version, extensions disponibles, etc.) [Fig.6].

> Nœuds

Comme souvent dans une architecture REST, les web services Neo4j exploitent les différentes méthodes proposées par le protocole HTTP : GET pour la lecture de données, POST pour l'altération (création/mise à jour) et DELETE pour la suppression. La création d'un nœud disposant de la propriété `nom=paul` est réalisée de la manière suivante :

```
> curl -X POST -d '{"nom":"paul"}' -H Content-Type:application/json http://localhost:7474/db/data/node
```

La structure JSON retournée correspond à une représentation du nœud créé.

```
{
  "outgoing_relationships": "http://localhost:7474/db/data/node/1/relationships/out",
  "data": { "nom": "paul" },
  "traverse": "http://localhost:7474/db/data/node/1/traverse/{returnType}",
  "all_typed_relationships": "http://localhost:7474/db/data/node/1/relationships/all/{-list|&|types}"
}
```

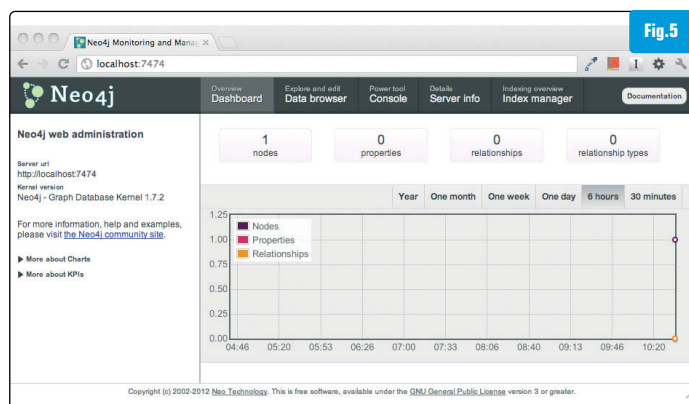


Fig. 5

```
/1/relationships/all/{-list|&|types}",
  "property": "http://localhost:7474/db/data/node/1/properties/{key}",
  "self": "http://localhost:7474/db/data/node/1",
  "properties": "http://localhost:7474/db/data/node/1/properties",
  "outgoing_typed_relationships": "http://localhost:7474/db/data/node/1/relationships/out/{-list|&|types}",
  "incoming_relationships": "http://localhost:7474/db/data/node/1/relationships/in",
  "extensions": {},
  "create_relationship": "http://localhost:7474/db/data/node/1/relationships",
  "paged_traverse": "http://localhost:7474/db/data/node/1/paged/traverse/{returnType}?pageSize,leaseTime",
  "all_relationships": "http://localhost:7474/db/data/node/1/relationships/all",
  "incoming_typed_relationships": "http://localhost:7474/db/data/node/1/relationships/in/{-list|&|types}"
}
```

L'attribut `self` (<http://localhost:7474/db/data/node/1>) est particulièrement intéressant : il s'agit de la référence au nœud. La valeur 1 située à la fin de l'url correspond à l'identifiant numérique unique du nœud au sein du graphe. D'autres attributs indiquent les url disponibles pour accéder aux différentes caractéristiques (propriétés, arcs, chemins) du nœud. A titre d'exemple, l'url <http://localhost:7474/db/data/node/1/properties> donne accès à toutes les propriétés ; <http://localhost:7474/db/data/node/1/properties/nom> ne retourne, quant à elle, que la valeur de la propriété `nom`.

```
> curl http://localhost:7474/db/data/node/1/properties/nom
«paul»
```

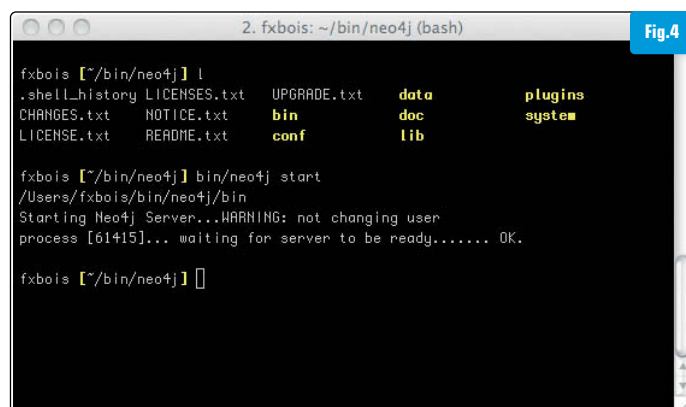


Fig. 4

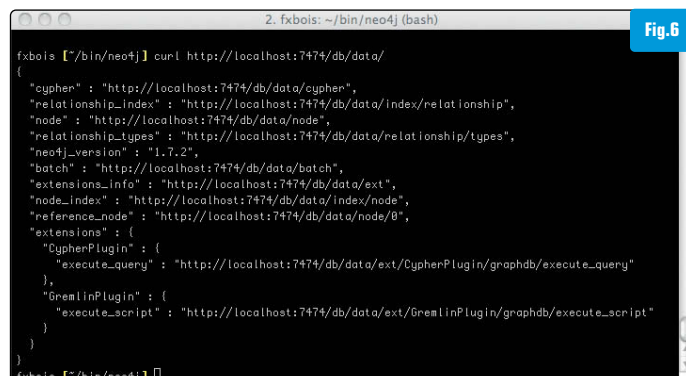


Fig. 6

> Arcs

La création d'un arc entre deux nœuds est réalisée en ajoutant un nouvel arc à la liste des arcs de l'un d'entre eux. Les arcs du nœud 1 sont représentés par l'URL suivante <http://localhost:7474/db/data/node/1/relationships>. Deux paramètres doivent être transmis : le nœud cible précisé avec le paramètre to, et le type de l'arc.

La requête ci-dessous relie le nœud 1 au nœud 2 par la relation « X connaît Y ».

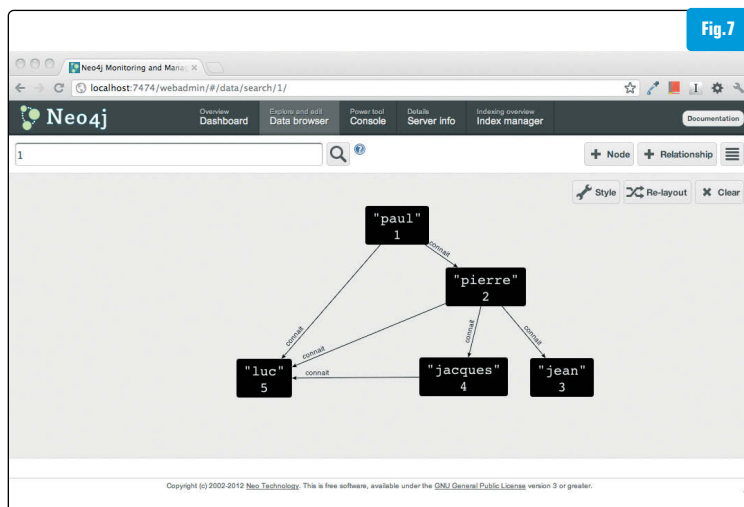
```
> curl -X POST -d '{«to»:»http://localhost:7474/db/data/node/2», «type»: «connaît»}' -H Content-Type:application/json http://localhost:7474/db/data/node/1/relationships

{
  «start» : «http://localhost:7474/db/data/node/1»,
  «data» : {},
  «self» : «http://localhost:7474/db/data/relationship/0»,
  «property» : «http://localhost:7474/db/data/relationship/0/properties/{key}»,
  «properties» : «http://localhost:7474/db/data/relationship/0/properties»,
  «type» : «connaît»,
  «extensions» : {},
  «end» : «http://localhost:7474/db/data/node/2»
}
```

Cette souplesse vis-à-vis du typage des arcs offre la possibilité de créer un réseau social (type « X est ami avec Y ») aussi simplement qu'un réseau professionnel de type LinkedIn (type « X collabore avec Y »). Il est également possible de faire coexister différents types d'arcs au sein d'un même graphe. Il devient alors possible de créer des graphes modélisant des connaissances plus vastes. Il s'agit précisément d'un sujet majeur sur lequel travaillent des sociétés telles que Google et Facebook. Qu'il s'agisse de l'Open Graph ou du Knowledge Graph, ces graphes cherchent à contenir une représentation universelle du savoir.

Il convient de noter que la rubrique Data browser du site permet de visualiser la structure du graphe (un paramétrage est nécessaire pour sélectionner les informations à afficher au niveau des nœuds) [Fig.7].

Le graphe présenté dans la [Fig.7] est celui utilisé pour illustrer les prochains exemples.



> Chemins et distance

Ajout de nœuds et création d'arcs correspondent aux opérations de base qui permettent de construire un graphe. L'intérêt de Neo4j se situe cependant ailleurs. La véritable complexité liée aux graphes réside dans les opérations mettant en œuvre les chemins entre les nœuds (il ne faut jamais oublier que les « boucles », source majeure de complexité, sont des phénomènes courants dans les graphes).

La distance entre deux nœuds est une information particulièrement intéressante dans le cadre d'un réseau social. Cette valeur est obtenue en cherchant le plus court chemin entre ces deux nœuds. Les chemins entre nœuds sont obtenus avec le web service http://localhost:7474/db/data/node/<ID_NOEUD>/paths

Les paramètres à transmettre sont les suivants :

- to : correspond au nœud de destination
- algorithm : nom de l'algorithme de parcours
- max_depth : longueur maximale du chemin
- relationships[«type»] : type d'arcs
- relationships[«direction»] : lorsque la valeur all est précisée, le « sens » des arcs n'importe pas

```
> curl -d '{«to»:»http://localhost:7474/db/data/node/4», «algorithm»:»shortestPath», «max_depth»:4, «relationships»: {«type»: «connaît», «direction»:»all»}}' -H Content-Type:application/json http://localhost:7474/db/data/node/1/paths

[ {
  «start» : «http://localhost:7474/db/data/node/1»,
  «nodes» : [ «http://localhost:7474/db/data/node/1»,
               «http://localhost:7474/db/data/node/2»,
               «http://localhost:7474/db/data/node/4» ],
  «length» : 2,
  «relationships» : [ «http://localhost:7474/db/data/relationship/0», «http://localhost:7474/db/data/relationship/2» ],
  «end» : «http://localhost:7474/db/data/node/4»
},
{
  «start» : «http://localhost:7474/db/data/node/1»,
  «nodes» : [ «http://localhost:7474/db/data/node/1»,
               «http://localhost:7474/db/data/node/5»,
               «http://localhost:7474/db/data/node/4» ],
  «length» : 2,
  «relationships» : [ «http://localhost:7474/db/data/relationship/5», «http://localhost:7474/db/data/relationship/4» ],
  «end» : «http://localhost:7474/db/data/node/4»
} ]
```

La réponse à ce type d'appel contient un tableau dont chaque cellule représente un chemin reliant Paul à Jacques. Les nœuds constituant chacun des chemins sont regroupés au sein de la cellule nodes

- Paul {1} > Pierre {2} > Jacques {4}
- Paul {1} > Luc {5} > Jacques {4}

La distance entre Paul et Jacques (égale à 2) est obtenue en accédant à la cellule length associée au premier élément du tableau réponse. Nous sommes sûrs de cette valeur car nous avons choisi l'algorithme shortestPath qui extrait les chemins de même longueur les plus courts.

Microsoft

tech·days

12,13,14 février 2013

DIGITAL IS BUSINESS

MICROSOFT VOUS SORT LE GRAND JEU
AU PALAIS DES CONGRÈS DE PARIS

3 PLÉNIÈRES
300 SESSIONS
17500 VISITEURS
140 EXPOSANTS
UNE TECHDAYS TV



Développeurs



Professionnels
de l'informatique



Décideurs
informatiques



Secteur public



Décideurs métiers

INSCRIVEZ-VOUS GRATUITEMENT SUR **WWW.MSTECHDAYS.FR**

En partenariat avec



Pour récupérer tous les chemins entre Paul [1] et Jean [3], nous modifions l'algorithme et choisissons `allPaths`.

```
> curl -d '{«to»:«http://localhost:7474/db/data/node/3», «algorithm»:«allPaths», «max_depth»:3, «relationships»: {«type»: «connait», «direction»:«all»}}' -H Content-Type:application/json http://localhost:7474/db/data/node/1/paths
```

```
{ {
  «start» : «http://localhost:7474/db/data/node/1»,
  «nodes» : [ «http://localhost:7474/db/data/node/1»,
               «http://localhost:7474/db/data/node/2»,
               «http://localhost:7474/db/data/node/3» ],
  «length» : 2,
  «relationships» : [ «http://localhost:7474/db/data/relationship/0»,
                      «http://localhost:7474/db/data/relationship/1» ],
  «end» : «http://localhost:7474/db/data/node/3»
},
{
  «start» : «http://localhost:7474/db/data/node/1»,
  «nodes» : [ «http://localhost:7474/db/data/node/1»,
               «http://localhost:7474/db/data/node/5»,
               «http://localhost:7474/db/data/node/2»,
               «http://localhost:7474/db/data/node/3» ],
  «length» : 3,
  «relationships» : [ «http://localhost:7474/db/data/relationship/5»,
                      «http://localhost:7474/db/data/relationship/3», «http://localhost:7474/db/data/relationship/1» ],
  «end» : «http://localhost:7474/db/data/node/3»
} ]
```

Deux chemins de longueur inférieure ou égale à 3 ont été trouvés :

- Distance 2 : Paul [1] > Pierre [2] > Jean [3]
- Distance 3 : Paul [1] > Luc [5] > Pierre [2] > Jean [3]

Il s'agit d'une autre fonctionnalité utile dans le cadre d'un réseau social : par l'intermédiaire de quels individus suis-je en relation avec une personne du réseau ?

> Cypher

Exprimer dans le langage courant une requête sur un graphe est généralement assez simple : « lister les amis d'amis d'un individu », « lister tous les amis communs entre deux individus », etc. Bien que Neo4j facilite les choses avec les API relatives au parcours des chemins, les développements nécessaires deviennent assez rapidement longs et fastidieux (boucles imbriquées avec gestion de « mémoire »). Pour faciliter la vie du développeur, la société Neo a élaboré le langage Cypher qui équivaut, pour les graphes, au SQL des bases de données.

Une requête Cypher est composée de 4 parties:

- **START** : quels sont les nœuds mis en œuvre dans ma requête (ex. nœud de départ)
- **MATCH** : la structure des chemins à étudier
- **WHERE** : les filtres sur les nœuds

- **RETURN** : les données à retourner

Illustrons la puissance de ce langage à l'aide d'un exemple qui a pour objectif de lister les noms des amis communs de Paul [1] et Jacques [4].

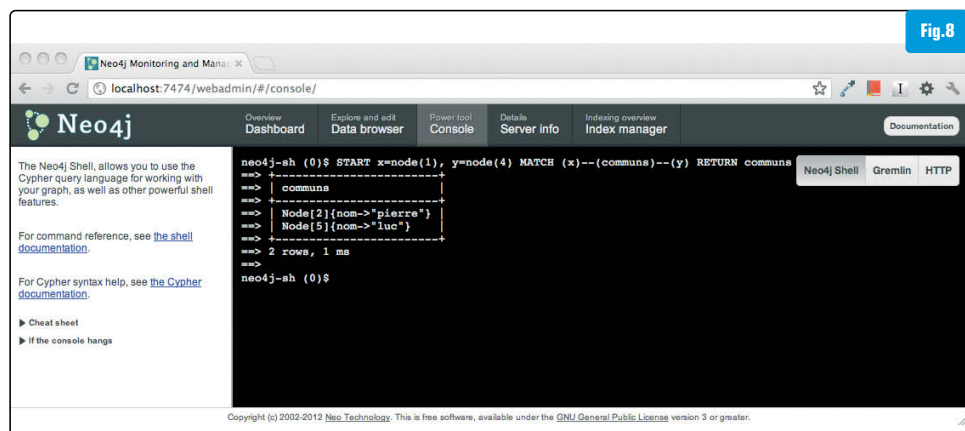
```
START x=node(1), y=node(4)
MATCH (x)--(communs)--(y)
RETURN communs.nom
```

Nous chargeons les deux nœuds dont nous avons besoin (nœud Paul et nœud Jacques), décrivons la « forme » du chemin qui nous intéresse (— signifie un arc) et utilisons la variable (communs) pour préciser les données que nous souhaitons obtenir dans le résultat. Une requête Cypher est transmise au serveur via le web service http://localhost:7474/db/data/ext/CypherPlugin/graphdb/execute_query en passant la requête dans le paramètre query.

```
> curl -d '{«query»: «START x=node(1), y=node(4) MATCH (x)--(communs)--(y) RETURN communs.nom}»}' -H Content-Type:application/json http://localhost:7474/db/data/ext/CypherPlugin/graphdb/execute_query
```

```
{
  «data» : [ [ «pierre» ], [ «luc» ] ],
  «columns» : [ «communs.nom» ]
}
```

L'interface web met à disposition une console permettant d'interpréter les requêtes Cypher [Fig.8].



> Conclusion

Tout comme les SGBD ont permis aux développeurs de s'affranchir des problématiques de stockage et restitution de données, Neo4j est en passe de prendre la même place dans le domaine de la gestion de graphes. En s'appuyant sur cette technologie aussi performante que scalable, le développeur est en mesure de concentrer son énergie sur les innovations fonctionnelles et de service.



François-Xavier Bois

Cofondateur de KerniX, François-Xavier Bois est en charge de l'innovation au sein de cette agence digitale parisienne. Il est l'auteur de plusieurs ouvrages consacrés aux technologies web (PHP, MySQL, WordPress).



NIDays

WORLDWIDE GRAPHICAL SYSTEM DESIGN
CONFERENCES

Le rendez-vous annuel des ingénieurs,
scientifiques et enseignants

**Au CNIT Paris La Défense,
le 12 février 2013**

Programme et inscription
gratuite sur www.nidays.fr



Il est temps, plus que jamais, d'innover pour relever les défis du monde moderne !

Vous préférez construire l'avenir plutôt que le subir ? Alors ne manquez surtout pas l'édition 2013 de NIDays et son lot d'innovations en matière de systèmes de mesure, de contrôle, d'automatisation et embarqués :

- ▶ 70 stands de démonstrations
- ▶ 2 conférences plénières, avec André Borschberg (Solar Impulse) en invité d'honneur
- ▶ 50 conférences techniques et métiers (dont 20 présentations d'utilisateurs)
- ▶ 14 sessions de TP pour s'initier aux matériels et aux logiciels
- ▶ 2 compétitions de robotique pour l'enseignement

Programme et inscription gratuite sur www.nidays.fr ▶ 01 57 66 24 24



Analysez vos données en Map Reduce avec Hadoop !

Analyser de gros volumes de données, indexer le web, ...On entend de plus en plus parler d'Hadoop pour ces sujets... De quoi s'agit-il exactement ? Et dans quels cas Hadoop est-il vraiment une solution intéressante ? Petite revue d'architecture...

En 2004, Jeffrey Dean et Sanjay Ghemawat de Google Labs publient *MapReduce: Simplified Data Processing on Large Clusters*,[1]

Doug Cutting, auteur de Lucene[2] et de Nutch[3], deux moteurs de recherche de la fondation Apache, décide d'implémenter MapReduce en tant que sous-projet de Nutch. L'idée était de faire de Nutch un moteur d'indexation distribuée, capable de gérer des volumes comparables à ceux de Google.

L'implémentation de MapReduce de Doug Cutting utilise alors NDFS (Nutch Distributed File System) pour le stockage distribué. Ce sous-projet de Nutch est basé sur une publication de 2003 de Sanjay Ghemawat, Howard Gobioff et Shun-Tak Leung de Google Labs : The Google File System[4].

Pendant deux ans, Doug Cutting et Michael J. Cafarella vont travailler à temps partiel sur ce qui va devenir Hadoop. En 2006, Doug Cutting rejoint Yahoo! pour travailler à plein temps sur MapReduce et NDFS avec une équipe dédiée. Par la suite, le projet devient projet officiel Apache, NDFS est renommé en HDFS (Hadoop File System) et le premier grand cluster Hadoop est mis en place chez Yahoo!

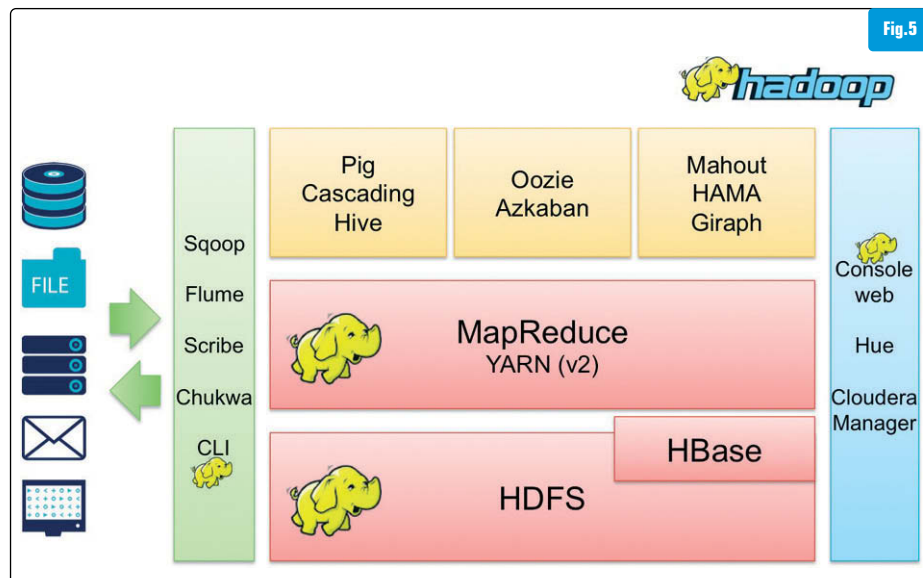
> Pourquoi Hadoop ?

Parce que le fils de Doug Cutting avait un élément qu'il appelait Hadoop.

Plus sérieusement, la question posée ici est celle de la raison d'être d'Hadoop.

Une technologie en soi ne sert à rien. Aussi est-il important de la replacer dans son contexte afin de bien comprendre à la fois sa raison d'être et ce que l'on peut raisonnablement en attendre.

L'un des éléments marquant de ces dernières années est l'explosion des volumes de données. Web, mobile, réseaux sociaux, de plus en plus de données sont échangées et les entreprises, emmenées par l'exemple des réussites de compagnies comme Google et Facebook prennent conscience de l'import-



tance de savoir traiter et faire parler ces données. Ce monde régi par l'analyse de téra voire de péta octets est celui du Big Data.

> Big Data ?

Economiquement, l'objectif du Big Data est de « tirer un avantage économique de l'analyse d'un grand volume de données ». Techniquement, lorsque l'on parle de Big Data, on atteint les limites de traitement des systèmes actuels.

Les 4 grandes classes d'architecture permettent d'illustrer ces limites. L'expérience montre qu'au delà de ces seuils, les systèmes traditionnels ne sont plus en mesure de fonctionner dans de bonnes conditions. Il faut alors recourir à d'autres solutions.

Des acteurs existent et proposent leurs solutions pour dépasser ces limites.

Hadoop apporte-t-il une réelle valeur dans ce cas ? On peut se poser la question. Hadoop se positionne sur trois des quatre limites des grandes classes d'architecture :

- Stockage distribué via HDFS
- XTP / eXtreme Transactions Processing via HBase
- Programmation parallèle via des frame-

works comme MapReduce

Hadoop se différencie par quelques points essentiels :

- Stocker un grand volume de données est peu coûteux
- Les données sont traitées brutes avec une qualité variable
- Il est facile d'ajouter de la capacité de calcul et de stockage
- Le système est ouvert

> Du stockage haute disponibilité accessible et performant ?

L'approche traditionnelle est de stocker ses données sur des systèmes chers, car équipés avec les meilleures technologies et sécurisés au maximum pour éviter toute perte d'un disque.

HDFS, à contrario, part du postulat que le matériel n'est pas fiable. Quoi qu'il arrive. A partir de ce cela, l'architecture d'HDFS est conçue pour pallier le manque de fiabilité du matériel et plusieurs choses en découlent :

- Favoriser un grand nombre de machines peu coûteuses, plutôt que quelques machines très chères. Cela permet en général de faire des économies

- Chaque fichier déposé sur HDFS est découpé en blocs de taille fixe et ces blocs sont répliqués plusieurs fois sur l'ensemble des machines

D'autre part, HDFS n'utilise pas le RAID mais dispose de son propre mécanisme de répliqua-tion au travers d'un large ensemble de machines (des nœuds) pour assurer la haute disponibilité et la résilience des données à faible coût.

Mais comment assurer la performance dans ces conditions ? L'astuce repose ici à la fois sur le mode d'accès aux données :

- un fichier est écrit une seule fois puis est accessible en lecture uniquement
- au sein de chaque nœud
- lors d'accès distant.

> La performance au sein de chaque nœud

Chaque nœud ne dispose pas d'un seul disque dur stockant tous ses blocs car cela conduirait à une saturation de la bande passante du disque dur (sur une machine entrée de gamme donc en SATA 150 en plus). Au lieu de cela, le stockage est réparti sur plusieurs disques durs, chacun disposant de son propre contrôleur et de sa bande passante dédiée. On parle d'une configuration JBOD.

Cela procure trois avantages :

- un disque plus lent sur un JBOD ne ralentira pas les autres
- à pleine utilisation, chaque disque peut individuellement être utilisé à son maximum
- si un disque tombe en panne, il ne gênera pas le fonctionnement des autres

> La performance lors des accès distants

Dans HDFS, une machine particulière appelée le NameNode est en charge de conserver les méta informations (noms des fichiers, des répertoires, taille, liste des blocs d'un fichier, ...), d'orchestrer les repli-

ca de blocs entre les machines et également d'indiquer à l'utilisateur (qui peut être un humain comme un programme), sur quelle machine il doit se connecter directement pour accéder à un bloc donné. Le fait d'utiliser ce chef d'orchestre permet d'assurer de bonnes performances, car les méta informations sont stockées en mémoire et la mémoire ne coûte pas cher. Pour les données qui prennent de la place, les blocs, l'utilisateur est redirigé directement vers le nœud les stockant. Cela signifie que HDFS distribue aussi la charge des connexions utilisateurs sur tous les nœuds du cluster. Ainsi, naviguer dans une arborescence HDFS revient à naviguer de nœud en nœud.

> Quel type de données dans Hadoop ?

Les données stockées dans HDFS ne sont pas soumises à un formatage particulier.

En fait, HDFS n'a aucune connaissance de ce qu'il stocke, il se contente de le stocker et de le répliquer. L'intérêt de cette approche est que cela permet à HDFS de rester simple sur cet aspect, les éventuelles contraintes de format provenant des frameworks utilisés pour le traitement de données.

> MapReduce, le premier framework d'Hadoop

MapReduce est un modèle de programmation parallèle issu de Google Labs pour l'analyse de données en batch. Dans Hadoop, il est implémenté sous forme d'un framework d'exécution en Java.

Pourquoi un framework Java ? La première raison est que Hadoop est en Java. MapReduce est donc codé dans le même langage que HDFS. Une raison plus profonde est que Java apporte plusieurs avantages non négligeables :

- gestion de la mémoire par la JVM
- portabilité, « Codez une fois, faites tourner partout »

- bonnes performances
- abondance de ressources maîtrisant le langage.

> A quoi sert MapReduce ?

MapReduce a été conçu pour remplir trois objectifs :

- fournir une API simple pour que les développeurs fassent du code distribué
- paralléliser en petits traitements unitaires de gros volumes de données
- déplacer le code vers la donnée plutôt que le contraire (colocalisation).

Ce dernier point est fondamental. C'est ce qui différencie le plus MapReduce des frameworks traditionnels de calcul distribué.

Le postulat de base de MapReduce est qu'il est moins coûteux de déplacer un code compilé sur les nœuds d'un cluster HDFS plutôt que les données vers le code compilé. Par conséquent, chaque nœud d'un cluster HDFS va faire tourner des JVM MapReduce qui traitent des blocs stockés sur ce nœud.

Concrètement, cela veut dire qu'un job MapReduce est un fichier JAR standard contenant un Main et implémentant deux sous-classes : Mapper et Reducer. Le tout est envoyé comme job au cluster Hadoop via le JobTracker.

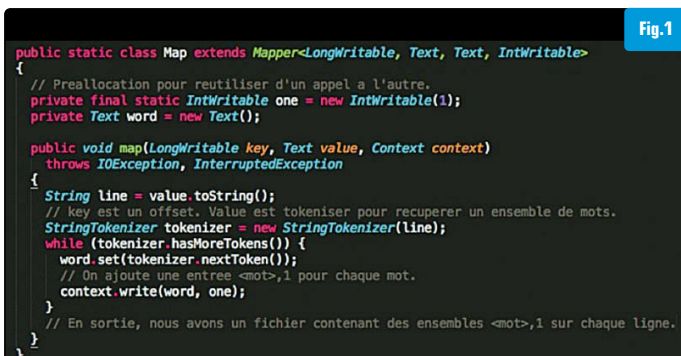
> Anatomie d'un job MapReduce

Un traitement MapReduce se découpe en trois phases distinctes :

- Le map
- Le shuffle & merge
- Le reduce

> La phase de map

A la soumission du job au cluster, un répertoire contenant les données d'entrée est indiqué. De l'ensemble des blocs de l'ensemble des fichiers de ce répertoire est calculé une liste de splits. Un split est en général le synonyme d'un bloc mais sa taille



```

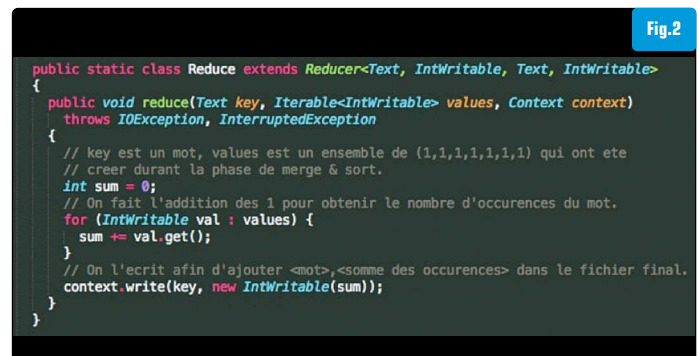
public static class Map extends Mapper<LongWritable, Text, Text, IntWritable>
{
    // Preallocation pour réutiliser d'un appel à l'autre.
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(LongWritable key, Text value, Context context)
        throws IOException, InterruptedException
    {
        String line = value.toString();
        // key est un offset. Value est tokeniser pour recuperer un ensemble de mots.
        StringTokenizer tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()) {
            word.set(tokenizer.nextToken());
            // On ajoute une entrée <mot>,1 pour chaque mot.
            context.write(word, one);
        }
        // En sortie, nous avons un fichier contenant des ensembles <mot>,1 sur chaque ligne.
    }
}

```

Fig.1

Exemple d'un mapper pour compter le nombre de mots.



```

public static class Reduce extends Reducer<Text, IntWritable, Text, IntWritable>
{
    public void reduce(Text key, Iterable<IntWritable> values, Context context)
        throws IOException, InterruptedException
    {
        // key est un mot, values est un ensemble de (1,1,1,1,1,1) qui ont été
        // créés durant la phase de merge & sort.
        int sum = 0;
        // On fait l'addition des 1 pour obtenir le nombre d'occurrences du mot.
        for (IntWritable val : values) {
            sum += val.get();
        }
        // On l'écrit afin d'ajouter <mot>,<somme des occurrences> dans le fichier final.
        context.write(key, new IntWritable(sum));
    }
}

```

Fig.2

Exemple d'un reducer pour compter le nombre de mots.

peut être configurée par l'utilisateur. Le nombre de splits correspond au nombre de JVM qui seront lancées pour traiter la phase de map.

Ensuite, la méthode map est appelée pour chaque record d'un split.

Un record est par défaut une ligne dans le split (au sens Unix du terme, '\n' étant le délimiteur). Là encore, la définition d'un record peut être configurée au lancement du job afin par exemple de gérer un ensemble de dictionnaires JSON ou un découpage plus complexe pour une donnée binaire (image, vidéo, ...). Les appels successifs à map() permettent de construire un fichier temporaire de clefs/valeurs appelé le résultat intermédiaire [Fig.1].

> La phase de shuffle & sort

Une fois que toutes les map sont terminées, la phase de shuffle & merge démarre.

Cette dernière va trier au moyen d'un quick-sort et regrouper les valeurs du résultat intermédiaire par clef. Puis, chaque clef est envoyée avec sa liste de valeur à une JVM de reduce. L'algorithme utilisé ici est appelé le HashPartitioner. Le principe est simple :

- le nombre de reducers est fixé au lancement du job
- chaque clef est hashée modulo le nombre de reducers

Cela permet de s'assurer qu'une même clef ne se retrouve pas sur deux reducers.

> La phase de reduce

Enfin, vient la dernière phase, le reduce. Une fois que les reducers ont reçu l'ensemble

des clefs à traiter, la méthode reduce() de l'interface Reducer est appelée pour chaque clef. A la fin de chaque appel, un record du résultat final est écrit. Chaque reducer va ensuite écrire ces records dans un fichier sur HDFS. Nous avons donc autant de fichiers de sortie que de reducers. Une fois le job terminé, l'utilisateur peut récupérer dans le répertoire de sortie, l'ensemble des fichiers écrits par les reducers [Fig.2].

> Comment lancer un job ?

Pour lancer un job, il faut une fonction main() dans la classe du job qui en configure les paramètres essentiels, tels que le type de données en entrée et en sortie, les classes de Map et de Reduce [Fig.3]. Ensuite, le job peut être lancé en le soumettant à un cluster Hadoop via la commande suivante : [Fig.4].

> Quels outils autour d'Hadoop ?

Hadoop possède un écosystème assez riche, fait d'éditeurs qui poussent leurs produits mais aussi et surtout de frameworks et de produits OpenSource venant étendre ses possibilités [Fig.5]. La future version stable d'Hadoop, YARN, va ouvrir de nouvelles possibilités en permettant à cet écosystème d'aller au-delà de MapReduce.

> Au-delà de MapReduce

MapReduce est un framework performant pour effectuer du traitement en batch sur des données pouvant être découpées en blocs indépendants. Par conséquent, MapReduce ne convient pas à tous les usages.

D'autres frameworks existent sur Hadoop pour mieux coller à des besoins spécifiques :

- HBase permet de mettre en place une base de données différente de ce qu'on connaît avec nos SGBD, car tournée vers la performance et ne respectant pas toutes les propriétés ACID[6]
- Giraph pour le graph processing
- Hama pour le calcul scientifique (Bulk Synchronous Parallel)
- Mahout pour le « Machine Learning »
- Cloudera Impala et Apache Drill, basés sur Dremel[5] de Google Labs permettent de faire des requêtes SQL distribuées sur de gros volumes de données

Bien que la plupart existent déjà sur Hadoop aujourd'hui, ces frameworks génèrent des requêtes MapReduce, ce qui en limite l'efficacité. Seul HBase a pris une voie un peu particulière mais cela pourrait faire l'objet d'un autre article.

La future version de Hadoop, actuellement en test, permet de s'affranchir de cette limitation en faisant de MapReduce un framework d'exécution parmi d'autres.

Il devient alors possible à d'autres frameworks spécialisés de devenir réellement efficaces et cela ouvre des portes pour le futur en positionnant Hadoop comme un socle de stockage et d'exécution distribuée, sans adhérence particulière à un mode d'utilisation.

Rémy Saissy

Consultant OCTO Technology

<http://www.octo.com>

<http://blog.octo.com>

Fig.3

```

public static void main(String[] args) throws Exception
{
    Configuration conf = new Configuration();
    Job job = new Job(conf, "wordcount");

    // On positionne les types du resultat.
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);

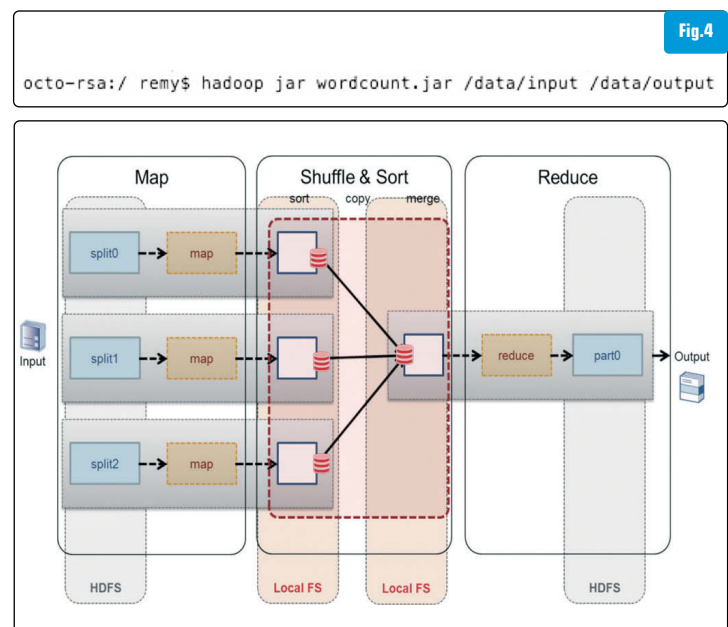
    // On indique quel est la classe du Mapper et celle du Reducer.
    job.setMapperClass(Map.class);
    job.setReducerClass(Reduce.class);

    // On indique quel est le formatage des fichiers d'entree et de sortie.
    job.setInputFormatClass(TextInputFormat.class);
    job.setOutputFormatClass(TextOutputFormat.class);

    // On dit au framework de lire le repertoire args[0] en entree
    // Et d'ecrire dans args[1] en sortie.
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));

    job.waitForCompletion(true);
}
    
```

Exemple du main() pour compter le nombre de mots.



OFFREZ-VOUS

pour cette nouvelle
année 2013

UN ABONNEMENT !

jusqu'à **-50%**

Code, gestion de projets, développement web, mobilité, Programmez ! est à la fois votre outil pratique, des articles de code par les meilleurs experts et votre veille technologique.



1

Abonnement 1 an au magazine

49 (au lieu de 65,45 €, prix au numéro)

2

Abonnement Intégral : 1 an au magazine + Archives Internet et PDF

59

3

Abonnement 2 ans au magazine

78 (au lieu de 130,90 €, prix au numéro)

4

Abonnement intégral 2 ans au magazine + Archives Internet et PDF

88



Toutes les offres en ligne : www.programmez.com

Abonnez-vous à partir de 3,80 € seulement par mois



Oui, je m'abonne

à retourner avec votre règlement à
Groupe GLI, 17 route des Boulangers 78926 Yvelines cedex 9

- ☐ **Abonnement 1 an au magazine** : 49 (au lieu de 65,45 €, prix au numéro)
- ☐ **Abonnement Intégral : 1 an au magazine + archives Internet et PDF** : 59 (au lieu de 65,45 €, prix au numéro)
- ☐ **Abonnement 2 ans au magazine** : 78 (au lieu de 130,90 €, prix au numéro)
- ☐ **Abonnement intégral 2 ans au magazine + archives Internet et PDF** : 88

☐ M. ☐ Mme ☐ Mlle Entreprise : _____ Fonction : _____

Prénom : _____ Nom : _____

Adresse : _____

Code postal : _____ Ville : _____

Tél : _____

E-mail : _____ @ _____

(Attention, e-mail indispensable pour les archives sur internet)

☐ Je joins mon règlement par chèque à l'ordre de Programmez ! ☐ Je souhaite régler à réception de facture

Android, iOS, Windows Phone 8 : Choisir sa plateforme de développement mobile

Quelques semaines après la disponibilité de Windows Phone 8, et pour le dernier numéro de l'année 2012, nous allons parler d'un sujet chaud pour le développeur : quel système mobile choisir pour développer ses applications ?

D'un côté, deux monstres dominent les débats sur le marché du smartphone : Android (Google) et iOS (Apple). De l'autre côté, très loin derrière, vous avez Windows Phone, BlackBerry, etc. Récemment, IDC a livré quelques chiffres montrant l'archi-domination sur les systèmes (estimations 2012) :

- **68,3 %** pour Android
- **18,5 %** pour iOS
- **4,7 %** pour BlackBerry OS
- **2,6 %** pour Windows Phone

Il faut dire que Android est présent sur plusieurs dizaines de modèles, contrairement à Apple qui est le seul à produire des terminaux iOS, la prouesse de la Pomme n'en est que plus remarquable. Microsoft ne peut que faire mieux avec Windows Phone 8, mais le chemin sera très long pour espérer titiller Apple, voire, Google.

Pour le 3e trimestre 2012, deux constructeurs dominent le marché des smartphones : Samsung (56 millions d'unités vendues) et Apple (27 millions d'unités vendues). Depuis un an, Nokia s'écroule et tombe à seulement 10 millions (la nouvelle gamme Lumia devrait, si tout va bien, améliorer cela). La plateforme iOS demeure la première pour le



modèle économique et les applications payantes mais Google arrive peu à peu à mieux monétiser sa plateforme.

3 larrons en 2013

Android et iOS seront donc sans conteste les premiers choix. Difficile de faire l'impasse. Le troisième système est certainement Windows Phone 8, même si son marché se limite à environ 3 %, actuellement. C'est le système qui devrait logiquement le plus progresser en 2013. Reste à étendre l'écosystème et à attirer les développeurs. Sans marché de masse, Windows Phone 8 n'intéressera pas le développeur et les éditeurs.

Toujours en 2013, il faudra suivre BlackBerry 10 qui sera dévoilé fin janvier. RIM joue avec BB 10 sa dernière carte, et même purement et simplement sa survie. Nous ne le mettons pas sur notre liste de choix à cause de sa non-disponibilité. Si BB 10 propose de belles choses, il risque tout de même de se faire rapidement laminer par Windows Phone 8.

Dans ce dossier spécial, nous allons vous faire partager des avis de développeurs, nous interroger sur comment choisir sa plateforme, les outils, sans oublier les bons conseils. A vous de jouer.

François Tonic

Quels critères pour quelle plateforme ?

L'ère des travailleurs rivaux à leur bureau 40 heures par semaine appartient désormais au passé. Aujourd'hui l'utilisation de tablettes et smartphones dans l'entreprise se présente comme une opportunité pour les employés, mais représente également un défi conséquent à relever pour les sociétés et comporte quelques risques qu'il faut savoir gérer. Indépendamment de la taille des structures, la gestion des problématiques informatiques et des questions de sécurité ne cesse de croître à mesure que les entreprises rendent leurs données métier accessibles sur les terminaux mobiles.

Avant de se lancer dans le développement d'applications mobiles, il importe de prendre en compte un certain nombre de considérations. En effet, chaque environnement informatique est différent avec ses besoins propres.

De quel budget dispose-t-on ? De quelles fonctionnalités nouvelles a-t-on besoin ? Sur quelle plateforme souhaite-t-on rendre disponible son application ? Souhaite-t-on permettre la pratique du *Bring Your Own Device* dans son entreprise, tendance récente de plus en plus répandue des employés qui utilisent leur appareil mobile personnel pour utiliser des services et ressources de l'entreprise comme la messagerie professionnelle, ou accéder à des serveurs de fichiers, des bases de données internes ? Les fonctionnalités natives sont-elles critiques, indispensables ? De combien de temps dispose-t-on pour déployer l'application ? Quelle est l'importance de la vitesse, de la performance ? L'application requiert-elle la fonctionnalité spécifique d'un appareil ? Quels langages de programmation sont maîtrisés par les ressources internes ? ...

Le choix de la plateforme de développement en elle-même soulève bon nombre d'interrogations.

Néanmoins, en les comparant les unes aux autres, trois approches distinctes de développement d'une application se profilent : 1) l'approche de développement en code natif, 2) opter pour une solution Flex / Air, 3) recourir à la technologie HTML5.

Chacune de ces trois options présente évidemment des avantages et des inconvénients. L'enjeu est de savoir identifier l'approche qui sera la plus profitable à son entreprise.

CODE NATIF ?

Le développement d'une application mobile en code natif présente des avantages significatifs. Cette méthode permet d'accéder en profondeur aux paramètres de l'appareil et d'optimiser de manière plus efficace la performance. Elle rend disponible certaines fonctionnalités qui ne sont pas accessibles aux plateformes Flex et HTML5. Ainsi pour les applications requérant des fonctionnalités très spécifiques, uniques, il n'y a pas d'alternative au développement en code natif.

Fiabilité, rapidité et puissance sont donc les maîtres mots. Mais il faut minorer cette performance par la dépendance : tout développement en code natif correspond à un développement spécifique pour un type d'appareil mobile en particulier et tout déploiement sur un autre type de mobile nécessite un nouveau développement. Il faut donc prévoir autant de (re)développements de l'application que de terminaux...

Parmi les avantages d'un développement en code natif, citons :

- les outils, le support et les meilleures pratiques fournis par les fabricants qui permettent d'accélérer les développements,
- les performances,
- l'accès à la plupart des fonctions des plateformes,
- des interfaces utilisateurs plus familières (en termes de look & feel)
- ...

FLEX / AIR

Lorsque Adobe a remis à l'Apache Software Foundation le code source de sa solution Flex¹, tous ses détracteurs ont hurlé à la



mort de Flash. Mais Flash est loin d'être mort ! Il est vrai que Adobe ralentit progressivement ses efforts et son support de Flash pour la mobilité, mais la plateforme Adobe Air rencontre toujours un succès croissant. Apache Flex est parfaitement efficace aussi bien pour des applications de poste de travail que pour des appareils mobiles (Android, iOS, ...), voire des applications TV. Son grand avantage reste la rapidité de mise en œuvre pour un aspect visuel séduisant.

Signalons encore que la plateforme continue malgré tout de s'enrichir, notamment avec une série d'extensions pour plus de fonctionnalités natives des terminaux mobiles pris individuellement.

Avec la donation de Flex à la fondation Apache, c'est juste une nouvelle ère qui commence. Le SDK Apache Flex sera plus lentement amélioré par Adobe, mais également par une communauté beaucoup plus large. Les nouvelles mises à jour logicielles d'Apache Flex SDK, d'Adobe AIR Runtime et d'Adobe Flash Builder confèrent une puissance à ce jour inégalée pour le développement multiplateforme. Ainsi, la dernière mise à jour d'Adobe AIR Runtime (3.5) comporte déjà le support iOS 6 SDK, et la nouvelle version d'Adobe Flash Builder (4.7) comporte des fonctions de simulation, test, débogage pour iOS *on-device*.

1. PC World, *Adobe Donates Flex to Apache* :

http://www.pcworld.com/article/244060/adobe_donates_flex_to_apache.html

HTML5 ?

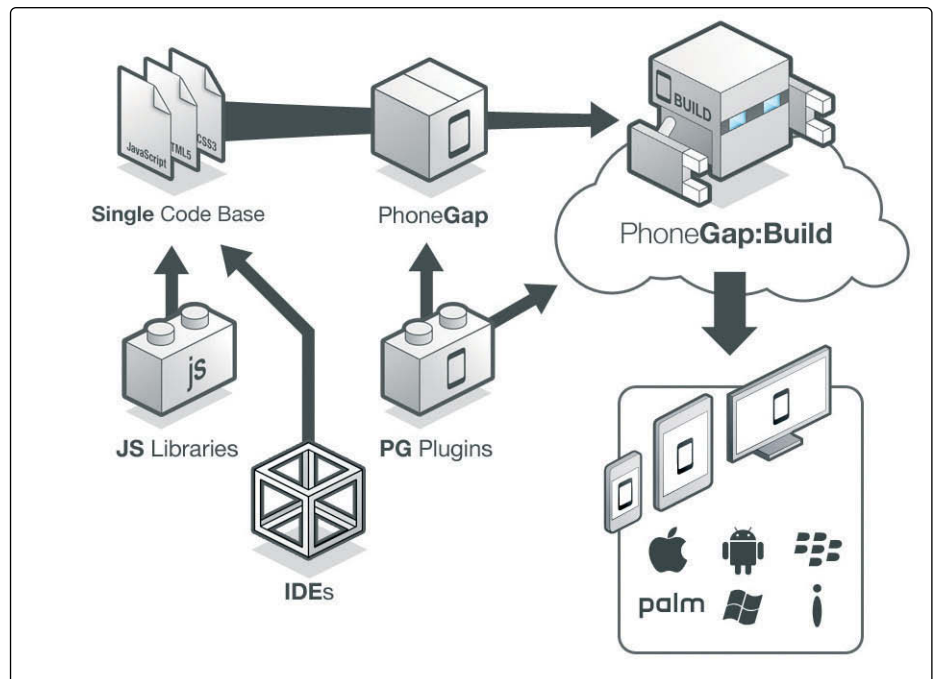
De nouveaux horizons pour le monde de l'application mobile ! L'évolution combinée du HTML, des feuilles de style CSS et de JavaScript est quand même phénoménale ! Les applications HTML5 sont accessibles depuis les navigateurs embarqués sur les tablettes et smartphones, mais peuvent également s'utiliser comme des applications natives via des frameworks hybrides. Un énorme pas en avant. Mais attention : HTML5 reste néanmoins dépendant et limité par les capacités propres de chaque navigateur et des Web-Kits (moteurs d'affichages conçus pour les navigateurs web mobiles).

Le HTML5 fournit aux développeurs des outils pour le stockage web hors connexion, des API pour la géolocalisation, des applications de dessin et bien d'autres encore...

Simultanément aux évolutions du HTML5 et aux derniers développements web, de nouveaux outils et frameworks plus complets et cohérents font leur apparition pour renforcer l'intégration HTML5 dans les navigateurs dédiés mobiles. On dénombre déjà plus d'une douzaine de frameworks disponibles sur le marché. Citons à titre d'exemple :

- JQuery Mobile
- Sencha Touch
- Backbone.js
- ...

Aujourd'hui, l'idée d'un développement mixte entre technologies web et technologies

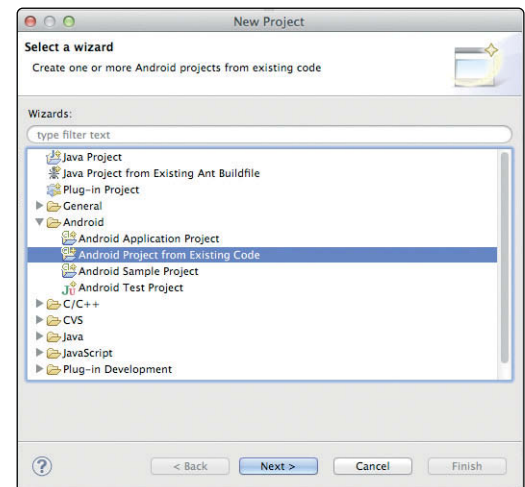


natives tend à se populariser. Généralement réparties selon leurs fonctions, les applications se créent désormais avec des nouveaux composants HTML plus rapides à construire. Elles sont réutilisables d'une plateforme à une autre et se mettent à jour de manière transparente pour l'utilisateur, sans processus d'autorisation, tout en préservant une expérience utilisateur de qualité. Prenons l'exemple de Phonegap. Certes, ce framework permet d'embarquer une page web HTML5/JS/CSS dans un shell natif, mais sa véritable force réside dans son potentiel hybride.

> Mais qu'est-ce qu'un framework hybride ?

Il s'agit de composants permettant l'affichage : tout démarre par un afficheur web. Cet afficheur doit pouvoir charger et interpréter du HTML (local ou distant), exécuter du JavaScript et des CSS. Puis permettre au JavaScript d'appeler des fonctions du code natif, et réciproquement. Toutes les plateformes le permettent aujourd'hui, mais malheureusement, sans unité et de façon trop hétéroclite.

Ainsi, les applications hybrides HTML5 se présentent comme un compromis intéressant parce qu'elles assurent une compatibilité entre les différentes plateformes de développement (dédiées au HTML5) et parce qu'elles peuvent utiliser les fonctionnalités matérielles et logicielles propres des



téléphones et des tablettes mobiles. Le choix d'une solution HTML5 développée sur une plateforme PhoneGap, Titanium ou Sencha, par exemple, vous donne accès aux API des constructeurs et vous ouvre les portes des App. stores.?

CONCLUSION

La plateforme de développement mobile parfaite n'existe pas, mais il est possible d'opter pour celle qui répond le mieux à vos objectifs business. Il y a beaucoup de questions à se poser, et autant de réponses associées. Enfin, au moment de se lancer, il faut voir les choses en grand, mais démarrer modestement !

Hans Verreydt,
Consultant, EoZen, Groupe SQLI
Traduction de l'article de Hans Verreydt : *Choosing your mobile development platform.*



Android, iOS, Windows Phone 8 : les développeurs parlent aux développeurs

Nous avons demandé à notre équipe de développeurs de partager leur retour d'expérience sur les trois plateformes majeures du développement mobile : iOS, Android et Windows Phone. Ils reviendront dans un premier temps sur les bases du langage de chaque OS et donneront ensuite leur analyse sur les outils de développement et de création d'interface graphiques proposés.

Nos experts développeurs

• iOS : Vincent **Saluzzo** • Android : Marc **Duvignaud** • Windows Phone : Guillaume **Brout** & Fitzgerald **Muiseroux**

iOS : Apple mise sur sa plateforme, son écosystème, sa maîtrise matérielle - logicielle



LANGAGE

La difficulté principale du développement iOS réside dans l'adaptation à un langage peu démocratisé avant l'arrivée de l'OS d'Apple : **Objective-C** (langage historique de NeXT puis de OS X, dérivé du C, NDLR). Non pas que le langage soit compliqué en lui-même, mais il dispose d'une syntaxe et d'un fonctionnement spécifique qui peut vite devenir déroutant si on ne possède pas de bonnes bases.

> Les bases du langage

Prenons l'exemple de création d'une classe en Objective-C :

```
@interface Article : NSObject {
    // iVar
}

// Propriété et méthodes
@end
```

Une implémentation s'écrit de la sorte :

```
@implementation Article

@end
```

Jusque-là rien de bien compliqué si ce n'est qu'ici, interface désigne le prototype d'une classe et non une interface comme en Java, qui elle est représentée en Objective-C par le mot clé : @protocol, exemple :

```
@protocol ArticleProtocol {
-(void) parseText:(NSString*)content
-(NSDictionary*) mapContent;
@end
```

Dans la déclaration d'une interface, il y a deux types de variables : les iVar et les Propriétés. Pour faire simple, les iVar déclarées dans une classe ne sont accessibles qu'à l'intérieur de celle-ci. Une propriété (@property) rend accessible la variable depuis l'extérieur (génère

automatiquement des getter/setter). C'est la version simplifiée de la différence entre un iVar et une propriété.

En réalité, Objective-C est un langage qui respecte totalement le Key-Value Coding (KVC). Cela veut dire qu'un objet quelconque peut être traité comme un dictionnaire.

Par exemple :

```
monArticle.title = @"Un Article Super !";
```

qui permet de modifier le contenu de la propriété title pourrait être écrit d'une autre manière :

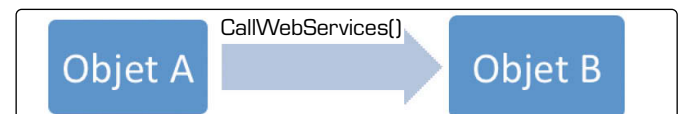
```
[monArticle setValue:@"Un Article Super!" forKey:@"title"];
```

Autrement dit, le langage permet un développement très dynamique car si la propriété «**title**» n'existe pas, nous pouvons la rajouter directement depuis l'exécution du programme (et non pas à la compilation) et accéder à cette propriété de la sorte.

Une des subtilités les plus importantes de l'Objective-C, c'est le **système de message**.

En Objective-C, le développeur n'appelle que très rarement des fonctions dans son code, il envoie des « messages ». Pour celui qui ne le sait pas, cela ne change pas grand-chose de l'appel de fonction, pourtant une différence est majeure ici :

L'objet qui envoie le message ne peut s'assurer que celui-ci arrivera à destination. Pour cause, les objets ne discutent que très peu directement, ils passent par un intermédiaire :



Se traduit en Objective-C par [fig.1].

Dans la plupart des cas, cela ne change rien pour le développeur, si ce n'est qu'il doit prêter attention sur les méthodes implémentées dans l'Objet B, le code compilera si l'objet B n'implémente pas la méthode appelée par l'objet A. Cependant cela générera une exception à l'exécution.

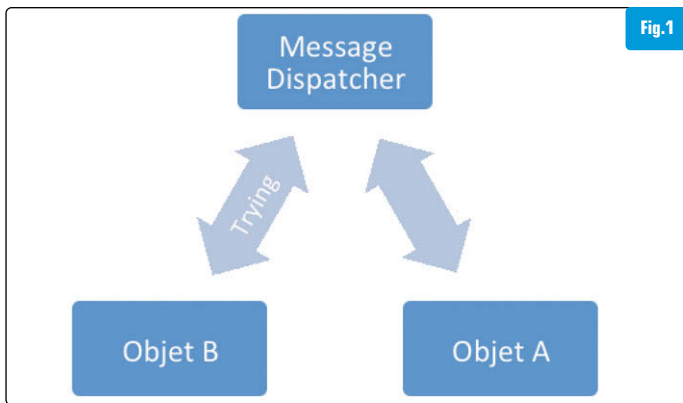


Fig.1

L'avantage réside plus dans les possibilités que cela apporte. Il est plus facile de développer un pont entre l'Objective-C et un autre langage, ou faciliter la sérialisation de données (exemple : CoreData) avec un système de message (runtime) qu'un appel de fonction (compilation time). L'envoi de message diffère de la syntaxe classique d'appel de fonction :

En Objective-C :

```
[Article callWebServices];
```

En C++ :

```
Article->callWebServices();
```

Bien entendu, une syntaxe d'un langage s'apprend avec le temps, il suffit de manipuler un peu un programme écrit en Objective-C pour très vite s'adapter à celle-ci.

> L'Objective C hérite complètement du C

L'avantage du langage Objective-C par rapport au C# ou au Java, c'est qu'il n'est qu'un superset du langage ANSI C. Autrement dit, toutes les bibliothèques créées et compilées avec le langage ANSI C sont exploitables depuis un programme Objective-C. De ce fait, le compilateur Objective-C (LLVM ou GCC-OBJC) peut compiler du code C ne contenant aucune ligne de code en Objective-C.

Un programme dédié à iOS doit néanmoins contenir un minimum de code source écrit en Objective-C, car un programme iOS s'appuie sur un framework nécessaire au développement d'application iOS: Cocoa Touch (ou UIKit). Celui-ci permet l'implémentation d'interfaces graphiques et la gestion des Entrées/Sorties standards d'une application iOS. Grâce à cet héritage, il est possible d'utiliser des bibliothèques standards écrites en C: libxml, libz, etc.

On aime

Langage ultra-dynamique
Ecriture de programme en C
Réutilisation de bibliothèques standards du C

On aime moins

Syntaxe particulière

> API : des frameworks dédiés

On trouve beaucoup de bibliothèques dans les outils de développement fournis par Apple, notamment les frameworks disponibles dans le SDK. Certaines de ces bibliothèques permettent d'effectuer des traitements très compliqués nécessaires à des applications complexes :

- OpenGL.framework - Dédié à l'utilisation d'OpenGL ES dans une application iOS
- OpenAL.framework - Dédié à l'utilisation d'Open AL dans une application iOS

- CFNetwork.framework - Permet l'utilisation des interfaces réseau de manière précise (Socket BSD, etc.)
- ImageIO.framework - Permet l'analyse, la création et la lecture d'image

Cependant, il existe aussi des bibliothèques qui permettent de faciliter l'implémentation d'un grand nombre des besoins logiciels pour une application standard :

- CoreGraphics.framework - permet l'utilisation de contextes graphiques (pour le dessin notamment)
- MessageUI.framework - permet la création de fenêtres, dédié à l'envoi de messages (SMS, iMessage, Mail)
- MapKit.framework - permet l'implémentation de cartes géographiques au sein d'une application
- MediaPlayer.framework - permet la lecture de médias dans une application

La plupart de ces bibliothèques (excepté les CoreLibraries), s'appuient sur d'autres bibliothèques pour faciliter le travail du développeur. De plus, il est très facile de trouver des bibliothèques Open Source pour réaliser un composant visuel (ou autre) dans une application. Un site internet se propose de référencer ces bibliothèques : www.cocoacontrols.com

On aime

Bibliothèque standard très efficace
Forte communauté Open Source

On aime moins

IDE

Il n'est pas possible de réaliser une application iOS sans utiliser XCode (des outils tiers permettent tout de même de générer in fine du code Objective-C, NDLR). Ce n'est ni plus ni moins que l'environnement de développement intégré fourni par Apple.

Apple mise toujours sur l'ergonomie de ses applications, cela s'applique aussi à l'environnement de développement proposé : XCode. Les menus sont bien pensés et intuitifs. Il est juste regrettable qu'Apple ne propose pas la personnalisation de cet outil, à la manière d'Eclipse ou de Visual Studio qui permettent de déplacer l'ensemble des fenêtres.

Néanmoins, Apple nous facilite énormément la création d'interfaces utilisateurs directement depuis XCode. L'outil Interface Builder permet de créer des interfaces graphiques aisément, de les personnaliser et d'intégrer le comportement de celles-ci au code source via des drag&drop [Fig.2].

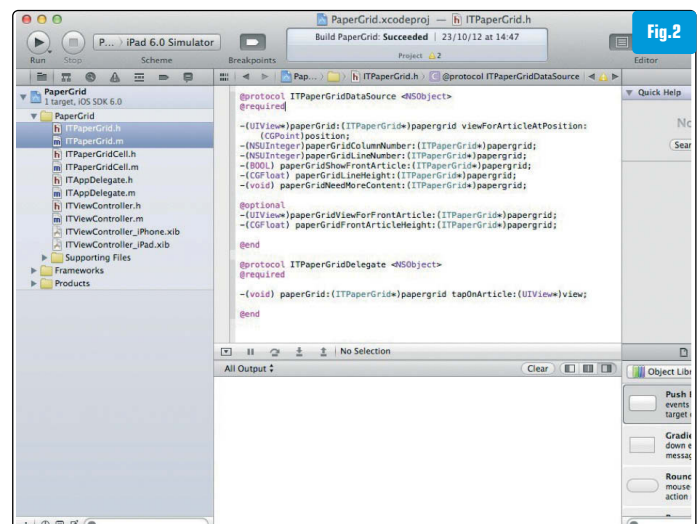


Fig.2

On aime	On aime moins
Ergonomie	Peu (ou pas) de personnalisation possible

> Intégration du versionning: Git

Apple fournit un outil de versionning intégré dans XCode pour assurer un versionning local. Celui-ci s'appuie sur Git. L'avantage c'est que Git est à la mode en ce moment, notamment grâce à sa démocratisation via différents projets: GitHub, Gitorious, etc. Cependant, la gestion de Git par Apple laisse à désirer lorsque celle-ci s'intègre avec des repository distants tels que GitHub. Dans ce genre de situation, il est préférable d'utiliser la ligne de commande pour gérer le versionning distant. Le versionning dans XCode permet d'avoir un aperçu de notre projet réalisé il y a quelques minutes, quelques heures, voire quelques jours et de se rendre compte des modifications effectuées, ce qui permet très facilement d'observer les possibles erreurs ajoutées, ou les morceaux de code importants supprimés par erreur... A noter que l'outil est assez peu adapté pour le travail en équipe, il arrive fréquemment de rencontrer des problèmes à l'utilisation.

On aime	On aime moins
Simplifie l'utilisation de Git	Peu recommandé dans le cadre d'utilisation d'un repository distant
Versionning local de tout le projet	Peu pratique en équipe

> Le debugging

XCode fournit deux niveaux différents d'outils pour le debug des applications :

- Un débogueur pas à pas classique
- Instruments : Suite d'outil d'analyse d'un programme.

Outre le fait que le débogueur soit complètement intégré à l'environnement de développement, l'utilisation d'instruments permet de détecter des failles de mémoires ou des blocages de threads à des points insoupçonnables. Apple fournit aussi un outil d'automatisation des événements utilisateurs, permettant de répéter x fois un tap sur l'écran à un endroit précis et bien d'autres interactions. Ce genre d'outil permet de créer des bench test aisément...

A noter que le débogueur a parfois du mal à correctement analyser les objets et certaines exceptions sont difficiles à déboguer car XCode ne nous montre pas toujours où l'exception est levée.

On aime	On aime moins
Instruments très pratique	Débogueur assez instable
	Difficile de relever d'où est levée une exception

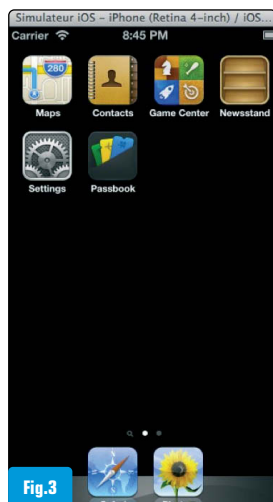


Fig.3

> iOS Simulator

L'un des points clés de XCode est son simulateur iOS. Celui-ci est plus qu'abouti en termes de rendu et de manipulation. Bien qu'il soit impossible de simuler des gestes complexes telles que le flip, le swipe ou autres, celui-ci capte complètement le pinch, le tap et le double-tap. De plus, il est possible de tester plusieurs versions d'iOS sur différents simulateurs matériels à la volée (iPad, iPad Retina, iPhone Retina, iPhone et iPhone 5). Contrairement au simulateur Android,

le simulateur iOS permet complètement de tester des programmes utilisant intensément la librairie OpenGL ES [Fig.3].

On aime	On aime moins
simulateur rapide et efficace	
Compatible avec les applications utilisant Open GL ES	

INTERFACE UTILISATEUR

> Les interfaces utilisateurs

La création d'interfaces utilisateurs sous XCode est largement simplifiée par l'outil Interface Builder. Celui-ci permet de n'écrire aucune ligne de code sur la mise en place des éléments graphiques d'une fenêtre. Il génère automatiquement des fichiers XIB qui sont des fichiers de description d'interface graphique au format XML. Ceux-ci sont transformés en fichier NIB qui sont au format binaire pour éviter d'être modifiés ultérieurement, et réduire l'empreinte mémoire qu'un format XML peut générer.

Interface Builder est totalement drag&drop. Chaque élément est inséré dans une vue par un glissé-déposé, ainsi que la configuration des interactions entre le code source et l'interface graphique.

iOS étant destiné à être totalement «MVC», la plupart du temps, une vue générée par l'Interface Builder d'XCode est associée à un fichier controller faisant le lien entre le code source métier, et l'interface graphique. Cependant, dans le cadre d'interface graphique plus complexe, il est tout à fait possible de modifier ou créer une interface graphique avec du code source. Deux possibilités :

- Création complète d'une vue en code source
- Création partielle de la vue sur Interface Builder et personnalisation de celle-ci depuis le controller

L'outil interface builder est un outil puissant et complet qui possède quand même quelques limites, principalement sur les possibilités de personnaliser certains contrôles natifs.

Depuis iOS 5, Apple a innové en apportant une fonctionnalité très utile : le Storyboard. Derrière ce grand mot se cache en fait un outil simple permettant de déterminer le schéma d'enchaînement de chaque fenêtre et les différentes actions entre elles, sans écrire une seule ligne de code [Fig.4].

On aime	On aime moins
Simplicité d'utilisation	Peu de possibilités de modifier les styles des contrôles
Storyboard	

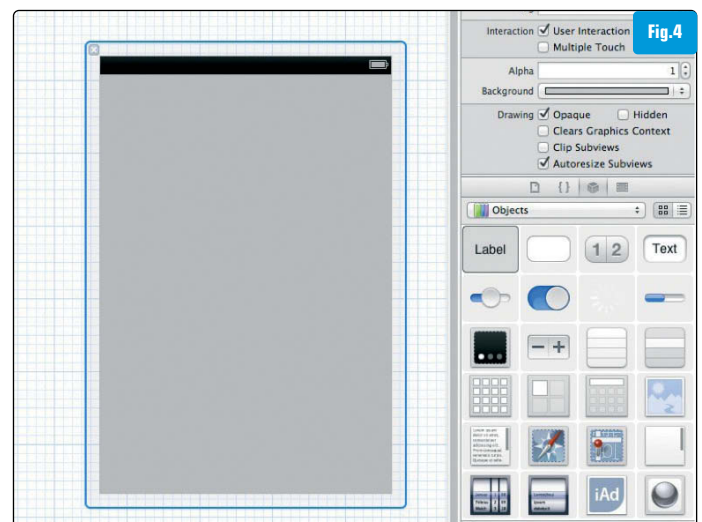


Fig.4



La rédaction : nous avons connu les premières versions d'Interface Builder (1999 – 2000) pour OS X 10 / 10.1, la version actuelle n'a plus rien à voir avec l'héritage OpenStep. L'outil suit scrupuleusement les guidelines d'interface Apple.

ANDROID :

Google domine

LANGAGE



La toute première chose à connaître lorsque l'on souhaite faire du développement d'application Android est le langage Java, choisi par Google pour créer des applications sur le système.

> Les Bases du langage

Contrairement à l'Objective-C, le langage Java est beaucoup plus classique et sera bien plus intuitif pour les développeurs ayant fait de la programmation orientée objet en C#, en VB ou en C++. Par exemple, en Java, une classe est créée tout simplement comme ceci :

```
public class Article {
    private String titre;

    public String getTitre()
    {
        return titre;
    }
    public void setTitre(String titre)
    {
        this.titre = titre;
    }
}
```

On peut remarquer dans le code ci-dessus l'un des principes de Java, appelé « encapsulation », qui est l'utilisation d'accesseurs (Getter / Setter) pour manipuler les attributs d'une classe.

Ainsi, pour récupérer ou modifier le titre d'un article il sera nécessaire d'utiliser les Getters/Setters et non pas directement les attributs :

```
public void EcrireArticle(String titreArticle)
{
    Article monArticle = new Article();
    monArticle.setTitre(titreArticle);
    System.out.println(monArticle.getTitre());
}
```

On voit dans cet exemple que l'on a créé une fonction nommée DoStuff qui va permettre de créer un article, de lui définir un titre passé en paramètre de la fonction, puis d'afficher celui-ci sur la console. Afin d'appeler cette méthode, il suffit tout simplement de faire :

EcrireArticle("Article Android");

Cette syntaxe, qui doit être familière à beaucoup de développeurs, permet d'utiliser très facilement le langage Java et d'être rapidement productif.

On aime

On aime moins

Syntaxe facile d'utilisation

> L'API

Le développement Android se base sur une utilisation partielle de l'API de la plateforme Java Standard Edition, le « cœur » de Java (la partie permettant de créer les interfaces graphiques des applications de bureau ayant, par exemple, été enlevée). On retrouve, entre autres, les packages java.util, java.io, java.text ou encore java.net qui sont connus et utilisés par tous les développeurs Java. En plus de ce « cœur », Google a ajouté l'API d'Android afin de permettre la création d'applications et d'interagir avec les fonctionnalités du Smartphone ou de la tablette.

C'est grâce à cette librairie que nous allons pouvoir créer les bases d'une application Android en créant les « Activités » qui représentent les fenêtres, les composants (boutons, listes, labels, ...) ou encore mettre en place les transitions entre ces différents écrans (via ce qu'on appelle les « Intents »). Outre la création de l'interface utilisateur, l'API Android permet aussi d'interagir avec la partie hardware de l'appareil mobile. Elle permet par exemple d'accéder au GPS, à la caméra ou encore aux capteurs (gyroscope, accéléromètre, capteur de lumière, ...). Enfin, l'API permet aussi d'accéder au système de base de données implémenté dans Android, SQLite qui est une version allégée des bases de données SQL.

Maintenant que nous avons vu les bases des API utilisées afin de créer une application Android, nous allons nous intéresser aux outils utilisés pour le développement.

La rédaction : attention à la fragmentation des systèmes Android. Les anciennes versions sont très largement utilisées. Veillez aussi à considérer la dépréciation des fonctions et API.

IDE

Bien qu'il soit possible d'utiliser un simple éditeur de texte pour créer une application Android (pour les plus extrêmes), Google recommande l'utilisation de l'IDE connu de tous les développeurs Java, Eclipse. En effet, un plugin a été développé (nommé ADT – Android Developer Tools) afin de permettre la création d'application Android directement depuis Eclipse. Ce plugin permet, entre autres, d'intégrer un éditeur XML adapté pour les ressources Android, un créateur d'interface utilisateur ou encore un système d'exécution / debug d'applications sur l'émulateur ou sur un appareil physique.

Nous allons maintenant voir un peu plus en détail les différentes fonctionnalités intégrées à cet IDE.

> Généralités Eclipse

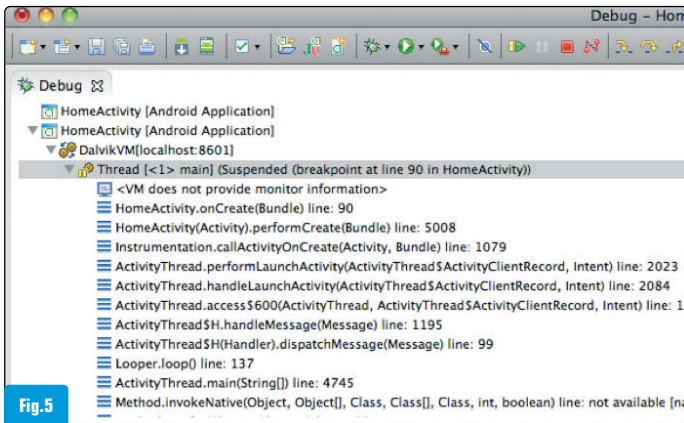
Le plugin ADT permet de parfaitement intégrer Android à l'IDE Eclipse et fournit les fonctionnalités habituellement demandées par un développeur, comme par exemple l'auto-complétion ou le quick-fix. Le plugin permet aussi de gérer les émulateurs Android et de choisir sur quel appareil exécuter l'application que l'on est en train de créer (émulateur ou appareil physique).

> Le debugging

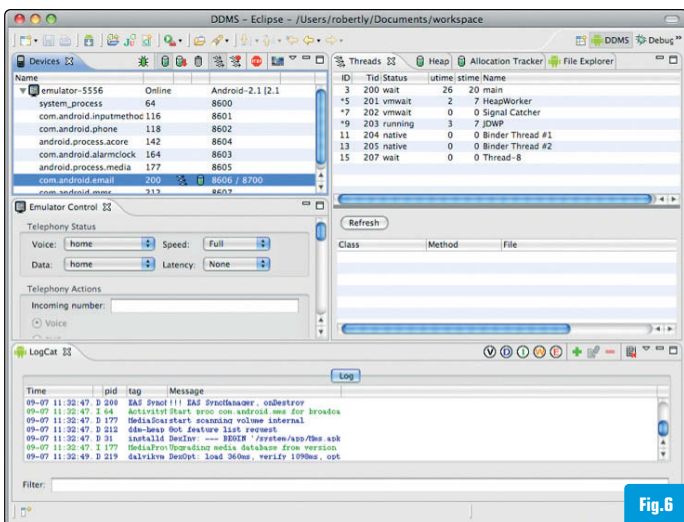
Le debug est une partie très importante de tout développement et est intégré directement dans l'IDE Eclipse. Lorsque l'on développe une application Android, le debugging est divisé en deux parties :

- Le debug classique : C'est la vue de debugging classique utilisée par tous les projets Eclipse (pas seulement les projets Android). Elle

permet de faire les « classiques » du debugging que l'on retrouve sur la majorité des IDE comme par exemple le debug pas à pas ou la consultation des threads en cours d'exécution (voir un exemple ci-dessous – source : <http://developer.android.com/tools/index.html>) [Fig.5].



- **DDMS (Dalvik Debug Monitor Server)** : Cet outil de debug spécifique à Android permet de faire du debugging un peu plus centré sur le mobile avec par exemple le monitoring de l'allocation mémoire, des threads ou encore de l'utilisation du réseau. Il permet aussi d'utiliser « LogCat » qui est une vue affichant tout le Log généré directement par l'application en cours d'exécution ou par le système Android. Enfin, DDMS permet aussi d'interagir avec l'émulateur Android en simulant un appel, un type de connexion réseau (3G, Edge, ...) ou encore la réception d'un SMS. Voici un exemple de DDMS ci-dessous (source : <http://developer.android.com/tools/debugging/ddms.html>) [Fig.6].



On aime

Debugging classique très complet
Intégration avec les appareils Android

On aime moins

Outils parfois complexes, difficiles à prendre en main

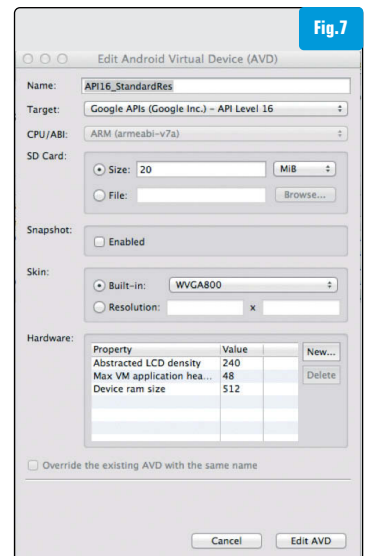
> Un émulateur encore faiblard

L'émulateur Android est l'un des points faibles du développement Android. En effet, il est très lent aussi bien au démarrage qu'à l'utilisation et un développeur perd énormément de productivité avec cet outil par rapport à l'utilisation d'un appareil physique.

Ce point faible tend néanmoins à se réduire avec l'arrivée de nou-

velles images système adaptées à l'architecture de l'ordinateur de développement (Intel a notamment sorti il y a peu une image pour la nouvelle version d'Android, Jelly Beans).

L'émulateur possède cependant un très gros avantage, il est possible de personnaliser énormément le système émulé (version Android, taille de l'écran, résolution, hardware,...) et donc de pouvoir tester sur les différents appareils Android présents sur le marché (qui sont très nombreux). Voici un exemple de la création d'un système émulé :



On aime

Personnalisation de l'appareil émulé

On aime moins

Lenteurs de l'émulateur

INTERFACE UTILISATEUR

Le plugin ADT a ajouté à Eclipse une partie permettant de créer des interfaces utilisateurs via un système de drag&drop comme on peut l'avoir sur les autres IDE.

Bien qu'il y ait eu des progrès, l'UI builder reste encore un peu trop basique et assez limitée lorsque l'on veut créer une interface un peu complexe et il est plus rapide d'éditer directement le XML de l'interface plutôt que d'utiliser l'outil de Drag&Drop.

WINDOWS PHONE 8 : Microsoft mise très gros

LANGAGE

Windows Phone 8 (WP8) est la dernière version de la plateforme de développement de Microsoft destinée aux Smartphones. Elle succède à Windows Phone 7.x. WP8 supporte les langages de développement de Microsoft C# et VB.NET, couplé à du XAML pour les interfaces graphiques, mais une des grosses nouveautés de WP8 est la possibilité de développer des composants en C++. WP8 supporte désormais 3 résolutions d'écrans : WVGA (800 x 480), WXVGA (1280 x 768), et 720p (1280 x 720).

Chose importante à noter, tout ce qui fonctionnait sous WP7.x continue à tourner sous WP8 : pas besoin de recompiler les applications.

Le mot d'ordre de toute application WP est performance, pour cela WP8 introduit le « Compile in the cloud ». Lorsque vous soumettez votre application, celle-ci est recompilée en utilisant NGEN (Native Image Generator), cela est également le cas désormais pour toutes les applications WP 7.x du store. L'utilisation de NGEN est possible puisque désormais les applications s'exécutent dans la Core CLR (comme sous Windows 8).



Le design pattern dominant dans le développement WP est MVVM (Model View View-Model). Ce design pattern contient différents degrés d'intégration, cependant il tend à supprimer le code behind afin de le déplacer dans un objet appelé View Model contenant toutes les données nécessaires au fonctionnement de notre page. Son avantage est de totalement diviser interface et la logique de l'application permettant au développeur et à l'intégrateur de travailler simultanément.

Il est possible de développer dans deux langages de programmation différents, C# et Visual Basic, les deux langages étant basés sur le Common language Runtime ils partagent les mêmes bibliothèques et sont totalement compatibles. Le C# étant le langage le plus répandu nous l'utiliserons pour la suite de l'article.

> Les bases du langage

Tout comme le JAVA, le langage C# hérite de la syntaxe C++, il ne dépaysera donc pas la majorité des développeurs.

Voici l'adaptation de la classe Article vue précédemment :

```
public class Article
{
    public String Titre {get; set;}
}
```

Le principe d'encapsulation en C# peut être généré automatiquement via la déclaration de propriété automatique grâce à la syntaxe {get; set;}. Cela génèrera les accesseurs (Getter / Setter) associés afin de manipuler les attributs d'une classe.

Ainsi, pour récupérer ou modifier le titre d'un article il est possible d'utiliser la propriété automatique et d'utiliser les accesseurs de manière transparente.

```
public void EcrireArticle(String titreArticle)
{
    Article monArticle = new Article();
    monArticle.Titre = titreArticle;
    Debug.WriteLine(monArticle.Titre);
}
```

L'avantage de C# par rapport au Langage de type Java est qu'il est beaucoup moins verbeux, cela amène une grande facilité de compréhension et un gain en productivité.

On aime

Syntaxe facile d'utilisation

Non Verbeux

On aime moins

L'API

Les développeurs WP disposent désormais de nouvelles API natives facilitant un peu plus le développement d'application et permettant d'interagir plus facilement avec le device (NFC, Achat In-App, Lock Screen...). Voici un schéma présentant l'API : [Fig.8].

WP8 supporte nativement le développement C++ et est capable de supporter une partie importante du SDK Windows 8, ce qui permet aux développeurs de partager une quantité non négligeable de code entre applications WP8 et Windows 8.

Rédaction : le SDK WP8 exige un Windows 64 bits. D'autre part, de nombreuses modifications existent entre le SDK 7.1 et le 8 (voir :

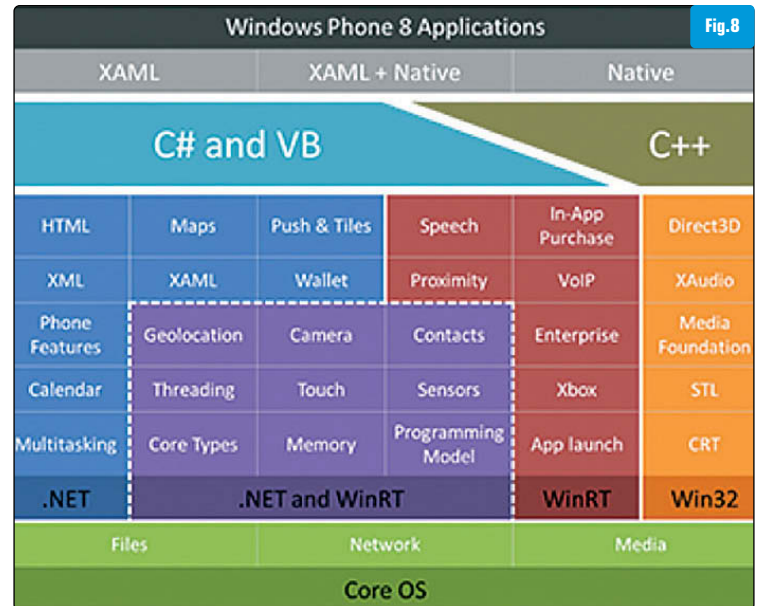


Fig.8

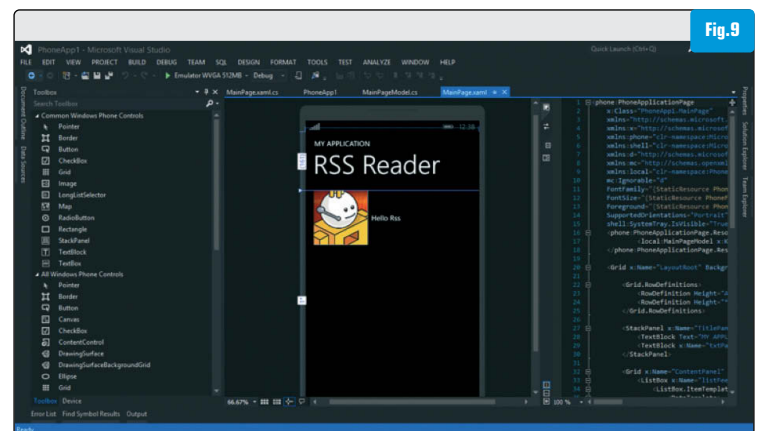


Fig.9

App platform compatibility for Windows Phone). Les applications WP 8 ne fonctionnent pas sur une ancienne version. Les applications produites par XNA Game Studio sont supportées mais impossible de créer de nouvelles applications WP 8 avec.

IDE

Le développement d'applications WP8 repose sur les outils de Microsoft, à savoir Visual Studio 2012. Le SDK WP8 est fourni avec une version gratuite (Express) de Visual Studio ou s'intègre à n'importe quelle version de celui-ci [Fig.9].

> Le debugging

Le debug s'effectue facilement grâce au débogueur de Visual Studio. A ce niveau-là WP8 bénéficie des années d'expérience de Microsoft et du .Net Framework et de la maturité de la gamme Visual Studio. Les fonctions de debugging vous permettent, par exemple, de vous arrêter à l'emplacement d'une procédure, d'inspecter la mémoire et d'inscrire des valeurs, d'observer le trafic des messages et d'examiner de plus près le fonctionnement de votre code.

Microsoft met également à disposition un mode de debug spécifique aux applications WP : Windows Phone Performance Analysis, permettant d'analyser précisément le comportement de vos applications [Fig.10]. De plus, vous pouvez également utiliser le Marketplace Test Kit, regroupant un ensemble de tests à valider avant de soumettre votre application sur la Store [Fig.11].

> L'émulateur

L'émulateur possède des fonctionnalités avancées qui permettent au développeur de tester l'application de la même manière que sur un téléphone, l'accéléromètre et la géolocalisation sont totalement simulables dans la section outils de l'émulateur. Il permet également de tester son application selon différentes configurations matérielles, telles que la quantité de mémoire disponible... [Fig.12]

On aime	On aime moins
La possibilité de customiser tous les contrôles	Pas possible de l'utiliser sur une machine virtuelle
Les dictionnaires de ressource	
Les samples datas	

INTERFACE UTILISATEUR

La gestion de l'interface utilisateur se fait soit avec les XAML design tools intégrés à Visual Studio, soit avec Blend (inclus dans Visual Studio 2012). Beaucoup plus de fonctionnalités sont désormais accessibles directement dans Visual Studio (Device Panel, Template editing, édition de contrôles...) Avec ces outils il est possible d'accéder à tous les styles des nombreux contrôles existants, soit en drag&drop, soit en éditant directement le xaml en vue splittée. Ils permettent aussi de créer des animations spécifiques (storyboards, visual states, transitions, easing). Blend offre en outre la possibilité de créer des dictionnaires de ressources (regroupant les éléments graphiques, styles, templates, behaviors) qui permettent de modifier très facilement et rapidement le rendu graphique d'un projet. L'un des plus gros avantages réside dans la forte interaction entre Blend et Visual Studio via les samples datas qui permettent d'afficher en mode design des éléments dynamiques générées par du code behind. Blend se révèle être un excellent outil de création d'UI, il requiert cependant une bonne pratique de l'outil pour être maîtrisé totalement [Fig.13].

On aime	On aime moins
La possibilité de customiser tous les contrôles	
Les dictionnaires de ressource	
Les samples datas	

CONCLUSION

Comme nous avons pu le voir, chaque plateforme possède un langage spécifique avec ses avantages et ses inconvénients, le choix du développeur sera donc fortement lié à ses compétences et son background. En ce qui concerne les outils de développement, Apple et Microsoft possèdent une longueur d'avance sur Android, en particulier parce que les outils de création d'interface sont beaucoup plus aboutis que sur Android. Le choix d'une plateforme par rapport à une autre doit donc se faire en intégrant d'autres facteurs que les aspects techniques : Quelle est la complexité de l'application ? Quelle population cible ? Quel budget pour le développement ? Quelle est la stratégie marketing associée ? Précisons enfin que pour Microsoft, Windows Phone 8 est l'opportunité de rattraper son retard sur la concurrence. L'attente est donc très forte concernant l'accueil que lui réserveront développeurs et utilisateurs.

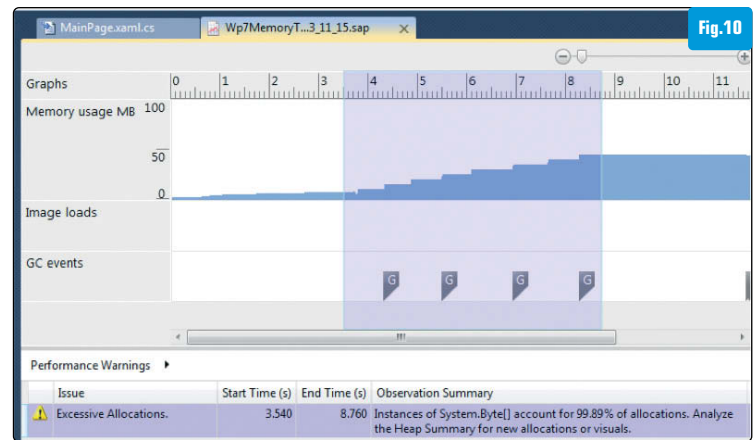
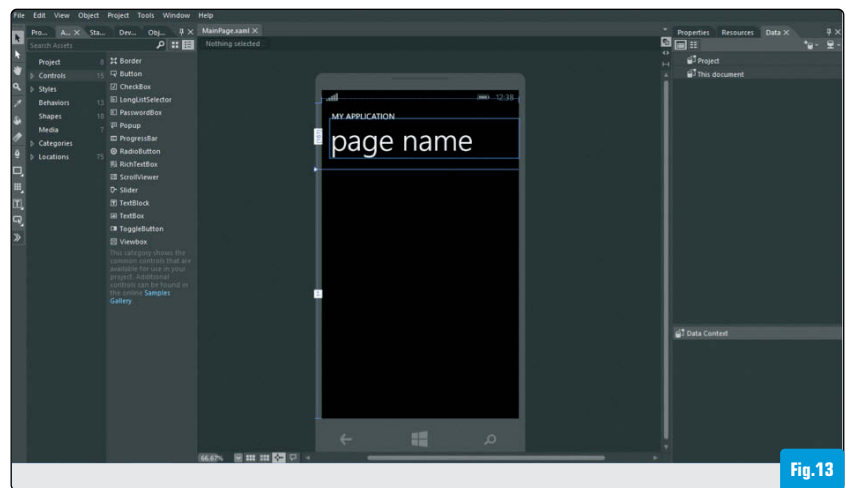
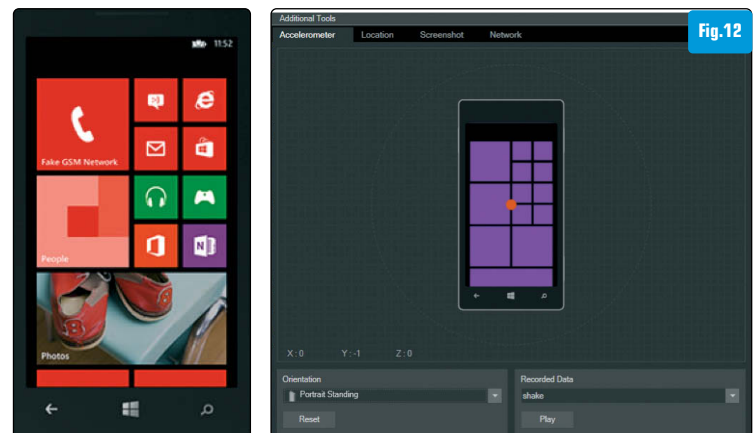


Fig.11

Result	Test Name	Test Description
Passed	Applicable Application Tile Images	<ul style="list-style-type: none"> View the Application list. Verify that the small mobile app tile image is representative of the application. From the Application list, tap and hold the small mobile app tile of your application and select 'pin to start'. Verify that the large mobile tile image on the Start screen is representative of the application. More info...
Passed	Multiple Devices Support	<ul style="list-style-type: none"> Install your application on two or more Windows Phone devices. Verify that the application can install and uninstall without error. After testing the above, ensure your application is installed, and launch it. Comprehensively test application functionality and features to verify that there are no device-specific issues. Verify that the application does not cause the device to stop responding or crash. More info...
	Application Closure	<ul style="list-style-type: none"> Launch your application.



5 leçons mobiles à méditer

Le développement smartphone est sans conteste le grand hype du microcosme des développeurs depuis 2 ans. Et il faut dire qu'il y a de bonnes raisons à cela : nouvelles technologies excitantes, croissance exponentielle des équipements et des applications, mais aussi quelques belles opportunités économiques. Reste à choisir la bonne technologie pour se lancer dans ce nouvel eldorado...

Dans le petit monde smartphone, 2 royaumes dominent très largement les autres : il s'agit bien évidemment d'Android et iPhone. De nombreuses études de marché - bien souvent contradictoires - sont disponibles sur le sujet et rendent plus ou moins bien compte du phénomène.

L'instrument de mesure qui semble le plus fiable pour comparer les technologies en termes de parts du marché provient des statistiques d'utilisation du web mobile.

En effet, cette utilisation reflète, bien plus par exemple que les ventes d'appareils, l'utilisation avancée d'un terminal smartphone et notamment la consommation d'applications.

Dès lors la domination absolue de la firme à la pomme et du petit robot vert ne fait guère de doute sur le marché français : [Fig.1]

En prenant un peu de recul, on peut d'ailleurs constater qu'au niveau mondial la réalité est un peu plus subtile, en particulier du fait du poids croissant des pays émergents moins denses en appareils haut de gamme [Fig.2].

Leçon 1 : si vous voulez toucher une part de marché significative, vous devez cibler iOS et/ou Android.

ANDROID VS IPHONE : CRITÈRES DE CHOIX POUR LE DÉVELOPPEUR

Reste à choisir entre ces deux univers. Et là deux philosophies radicalement différentes se font face. D'une part celle qu'Apple a toujours appliquée : un système fermé, dédié et optimisé pour une gamme d'appareils ciblée.

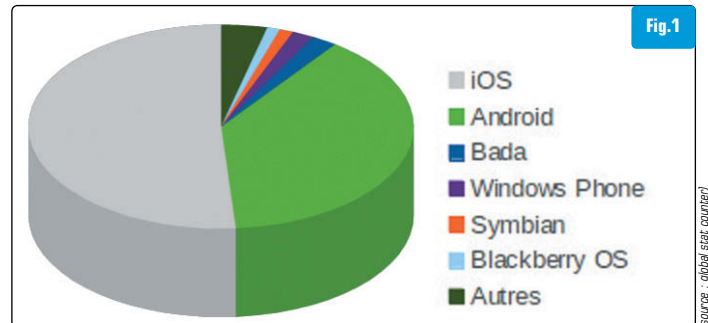
De l'autre, celle d'Android fortement influencée par le logiciel libre : un système ouvert, adaptable et applicable à un ensemble de devices potentiellement sans limite. Ces choix fondamentaux ont des conséquences à tous les niveaux pour les développeurs.

> Le langage

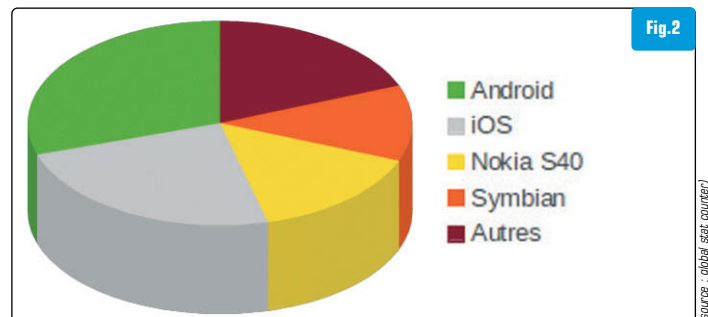
Le Java d'Android est certainement plus accessible à la majorité des développeurs français d'aujourd'hui que ne l'est l'objective-C d'Apple. Souvent plus verbeux, doté d'une syntaxe pouvant paraître déroutante et faisant appel à des connaissances parfois oubliées, et pourtant fondamentales ! comme les pointeurs ou la gestion de la mémoire, l'objective-C en fera abandonner plus d'un. Mais loin d'être un caprice de feu Steve Jobs, objective-C est un choix cohérent dans l'écosystème Apple où il est omniprésent et où ses qualités de langage "proche de l'OS" s'expriment au mieux.

A l'opposé, Java, très présent à la fois dans les études informatiques et les systèmes d'information français depuis une quinzaine d'années, saura séduire le plus grand nombre.

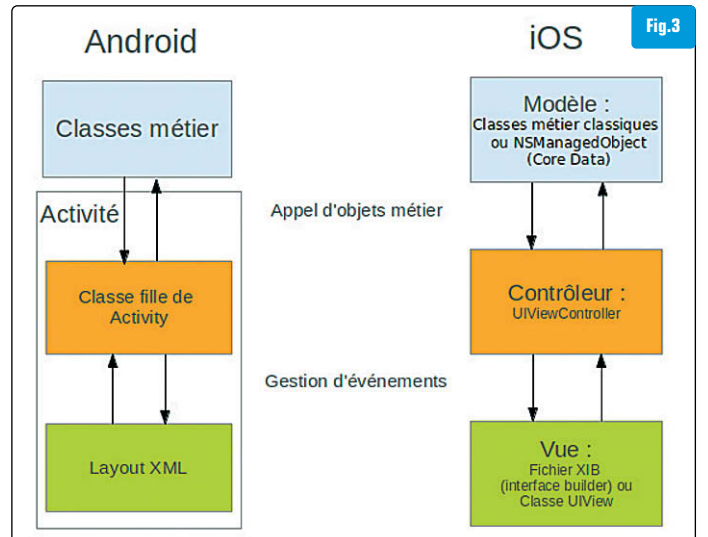
Et là encore, c'est un choix qui a du sembler naturel aux concepteurs d'Android : Java étant le langage par excellence pour déployer simplement des applications sur des architectures variées.



Statistiques d'utilisation smartphone sur le marché français



Statistiques d'utilisation smartphone sur le marché mondial



Architectures comparées d'applications sur iOS et Android

> Les kits de développement

On retrouve ces différences de conception dans les SDK proposés par chacune des deux plateformes. Alors que les classes et frameworks CocoaTouch d'Apple orientent la conception dans un cadre MVC relativement strict, l'architecture d'une application Android,

basée sur le concept d'activité, laisse une liberté à la fois attrayante et dangereuse sur des projets conséquents.

En prenant un peu de recul sur ces deux architectures, il est néanmoins possible de tracer un parallèle intéressant comme on peut le voir [Fig.3].

> Les environnements de développement

Pour fabriquer ces belles lignes de code, iOS ne vous donne guère le choix : il vous faut XCode... mais ne nous plaignons pas ! XCode est en effet gratuit et ne manque de rien pour vous permettre de coder de manière efficace. Mais la grosse contrainte, c'est que vous devez impérativement travailler sur Mac. Et pour cause : impossible de compiler le code sur une autre architecture.

Du côté d'Android, comme d'habitude, vous êtes libre. Eclipse reste souvent le premier choix, mais Netbeans ou IntelliJ IDEA sont des alternatives tout à fait productives. Là encore, tout l'outillage nécessaire est présent, avec néanmoins un bémol : l'émulateur de device Android dont les performances sont parfois proprement agaçantes.

> Performances et productivité

Les performances : voilà en effet un point sur lequel le développeur Android peut rencontrer des problèmes.

L'ubiquité de la machine virtuelle Dalvik ajoute un niveau d'abstraction qui éloigne le code applicatif du système d'exploitation : il en résulte des performances souvent inférieures à celles que le développeur iOS pourra atteindre avec son Objective-C, souvent plus laborieux dans un premier temps, mais finalement assez efficace sur la durée totale d'un projet.

Le garbage-collector de Dalvik permet certes de s'économiser la problématique de gestion de mémoire, mais il n'est pas rare de devoir revenir à une gestion manuelle de celle-ci.

Dans le même temps, Xcode commence à offrir un outil intéressant avec ARC (Automatic Reference Counting) qui permet théoriquement au développeur de se débarrasser de la gestion mémoire sur iOS.

Il est enfin encore un point qui rend les choses compliquées au développeur Android : la diversité des terminaux (plusieurs milliers !). Tous n'ont pas le même comportement par rapport à l'utilisation de telle ou telle API et la qualité du matériel est loin d'être toujours au rendez-vous... ce qui rend encore plus critique la question des performances ! C'est notamment pour cela qu'Android propose d'utiliser le Native Development Kit (NDK) pour développer en C/C++, au plus près du code machine.

L'utilisation du NDK exigera alors une compilation pour chaque architecture processeur ciblée.

> Déploiement des applications

Encore un point où nos deux philosophies s'affrontent. Le déploiement d'une application sous Android bénéficie d'une simplicité et d'une souplesse remarquables : il suffit de récupérer l'application sous forme de fichier apk pour l'installer sur un device.

Dès lors, les applis peuvent s'envoyer par mail, par USB ou être téléchargées via un serveur ou un "store privé". Bien entendu, la voie royale pour la distribution reste Google Play dont les procédures de publication sont relativement simples.

Déployer sur iOS est plus compliqué, même si la situation s'est, dans une certaine mesure, améliorée dernièrement grâce à des outils comme Diawi.

Apple limite en effet de manière considérable l'installation des applications, ce qui a par ailleurs quelques avantages, en particulier celui de la sécurité. Les méthodes légalement, c'est à dire hors "jailbreak", disponibles sont :

- l'App Store et sa procédure de validation très compliquée
- le déploiement ad-hoc sur des appareils déclarés auprès d'Apple
- le déploiement "entreprise"

Leçon 2 : *vous voulez rapidement bidouiller dans l'univers smartphone en ayant la possibilité de toucher à toutes les briques du système ? Choisissez Android. Vous voulez un résultat ultra-qualitatif pour un effort raisonnable sur le long terme ? Choisissez iOS.*

QUELLE PLACE POUR WINDOWS PHONE 8 ?

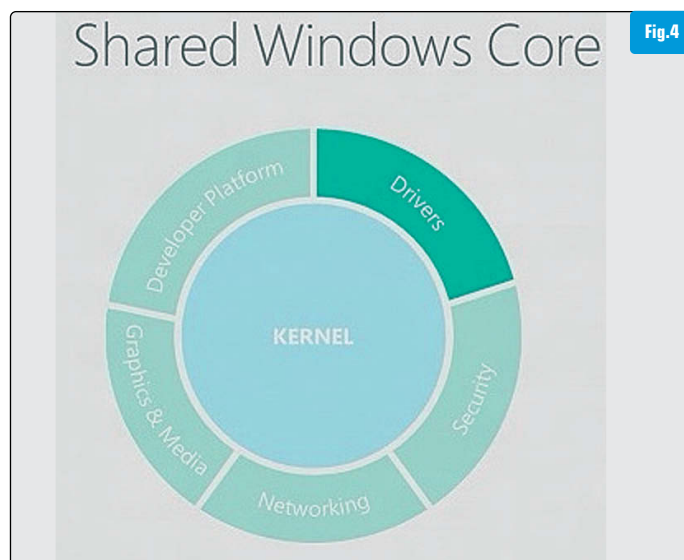
Si l'histoire est un éternel recommencement, Microsoft et ses tentatives de (re)prendre sa place dans l'univers du mobile en est une belle illustration. Souvenez-vous... En 2007, Microsoft domine le marché américain avec 42% des smartphones vendus tournant sur Windows Mobile quand Apple fait son entrée fracassante dans l'univers de la téléphonie.

Chutant à 27% de parts de marché dès l'année suivante, la firme de Redmond tente de réagir avec Windows mobile 6.5 en 2009 puis en 2010 avec Windows Phone 7 et le désormais défunt Kin.

Malgré le partenariat noué avec Nokia, la grande campagne de promotion de la gamme Lumia et la sortie de Windows Phone 7.5 en 2011, Microsoft ne parvient pas à percer. 2012 arrive, et là encore nouvelle stratégie : ce sera Windows Phone 8... qui casse la compatibilité avec Windows 7 comme ce dernier avait cassé la compatibilité avec la génération Windows Mobile.

Avec 2 refontes majeures en moins de 3 ans, l'historique n'a pas de quoi mettre le développeur en confiance.

Mais Steve Ballmer nous promet que "cette fois-ci, c'est la bonne" et le chambardement dans les différentes versions de Windows (8, RT, Phone 8, Server 2012) est tel que l'on peut au moins lui accorder le bénéfice du doute.



Le Share Windows Core

> L'architecture logicielle Windows Phone 8

La grosse nouveauté conceptuelle de Windows Phone 8, c'est le partage du noyau et de quelques fonctions de base avec Windows 8 : le Shared Windows Core.

Parmi celles-ci, on retiendra essentiellement le fait que Windows 8 et WP8 partagent désormais un nombre important de "bibliothèques de base" nécessaires aux applications. WP8 embarque en particulier la même base de runtime .NET que son grand frère et envoie définitivement les spécificités du "compact framework" aux oubliettes [Fig.4].

C'est d'ailleurs un point sur lequel les ingénieurs de Microsoft insistent particulièrement : la base du runtime .NET, le "coreCLR" est maintenant au coeur du fonctionnement des applications Windows Phone. Les optimisations apportées au coreCLR au niveau de la génération de code machine, de l'exploitation des processeurs multi-cœurs, et du garbage collector font, d'après ses évangélistes, de WP8 une vraie technologie de rupture en matière de performances des applications. Des propos qui restent à confirmer sur le terrain, mais les premières expériences semblent montrer que les choses avancent dans le bon sens sur cette question.

> Vue d'ensemble du SDK

Le SDK est disponible depuis fin octobre et propose gratuitement un ensemble comprenant :

- une version stand-alone de Visual Studio 2012
- Expression Blend pour Windows Phone
- toutes les bibliothèques nécessaires au développement d'applications WP 8 et 7.x avec du code .NET ou du code C++ "natif"
- XNA Game Studio pour la réalisation de jeux Windows Phone
- plusieurs émulateurs smartphone
- un client pour Team Foundation Server
- un ensemble d'outils de tests et de profilage des applications

Si le SDK est gratuit, il n'en reste pas moins que les conditions à remplir pour faire fonctionner la bête ne sont pas forcément accessibles à tous les développeurs : une machine fonctionnant impérativement sous Windows 8 64bits et disposant au minimum de 4Go Ram. Ces prérequis assez restrictifs rappellent singulièrement le modèle "iOS" qui a, semble-t-il, servi d'inspiration sur plusieurs points. Notons enfin que le SDK peut s'incorporer en tant que plugin à Visual Studio 2012.

> Développer pour Windows Phone 8

Le développement d'applications WP8 reprend les mêmes principes que le développement WP7 : soit XAML + C# ou VB, mais, et c'est encore une inspiration venue d'iOS et d'Android, il est désormais possible de développer en C++ natif. Ce nouveau mode de développement, outre un certain gain de performance, offre la possibilité d'utiliser de nombreuses bibliothèques C++ classiques telles que les moteurs Havok ou Unity et ouvre par exemple la porte à de vraies expériences de jeu avancées sur WP8.

Regardons donc comment se présentent les choses pour le développeur à travers un petit Hello World. Vous pouvez commencer par utiliser le designer d'interface pour placer un bouton dans notre application. VS2012 vous génère alors le code XAML correspondant :

```
<Button x:Name="Hello" Content="Hello" VerticalAlignment="Top"
HorizontalAlignment="Right"/>
```

Il vous suffit ensuite de double-cliquer sur le bouton pour créer l'événement "clic" sur le bouton. Un template du "code-behind" pour cet événement s'affiche dès lors :

```
namespace HelloWorld
{
    public partial class MainPage : PhoneApplicationPage
    {
        // Constructor
        public MainPage()
        {
            InitializeComponent();
        }

        private void Hello_Click(object sender, RoutedEventArgs e)
        {
        }
    }
}
```

Vous n'avez plus qu'à remplir le Hello_click avec le code adéquat. Par exemple :

```
MessageBox.Show("Hello World !", "Application de démo",
MessageBoxButton.OK);
```

Bref, rien de neuf pour les aspects basiques si vous êtes un habitué du développement Windows Phone 7 voire même du développement .Net en général. L'apprentissage sera plus à faire du côté des évolutions des API, ce qui tombe bien puisque la documentation est plutôt bien fournie, notamment en exemples.

> Tablettes et histoires de compatibilité

Les annonces relatives à la compatibilité des applications Windows Phone 8 avec le reste de l'écosystème Windows paraissent parfois contradictoires. Commençons par la compatibilité WP7, pas forcément inutile dans la mesure où les téléphones WP7 (et 7.5) ne pourront jamais être upgradés en WP8 (par contre l'upgrade en 7.8 sera possible) :

- une application WP7.x peut être upgradée en WP8 simplement, aux différences d'API près
- VS2012 permet le développement d'applications pour WP8 et/ou WP7.x Pour faire fonctionner une app sur les deux types de plateformes, il faudra là encore gérer les différences d'API

La grosse différence avec les univers Android et iOS, c'est l'absence de compatibilité des applications Windows Phone avec les nouvelles tablettes Windows. En effet, contrairement à ses concurrents, WP8 n'est pas un environnement destiné à ces dernières.

Pour rappel, la solution Microsoft pour tablettes se nomme Windows RT si elles fonctionnent sur architecture ARM (Nvidia, Qualcomm) ou tout simplement Windows 8 sur x86 (Intel, AMD). Comme nous l'avons vu plus haut, les systèmes sont proches et même s'ils ne sont pas directement interchangeables du point de vue des applications, il est possible de mutualiser une bonne partie de code source entre ces différents systèmes.

Cette mutualisation s'opère grâce aux "portable class libraries" (cf. Programmer novembre 2012) qui ne sont pas nouvelles mais qui

prennent tout leur sens avec le rapprochement des différentes versions de Windows.

On ne vous recommandera dès lors jamais assez de suivre avec attention les quelques recommandations de Microsoft sur l'architecture de vos applications (en particulier le fameux MVVM) si vous voulez en tirer parti. Quelques proofs of concept de conversation d'application d'un univers à l'autre s'avèrent assez bluffants... même si là encore, nous manquons de recul pour voir ce que cela pourrait donner sur de vrais projets.

Leçon 3 : *se lancer dans Windows Phone 8 représente un investissement dont la rentabilité est encore incertaine malgré des promesses qui paraissent alléchantes. Soyez prudents.*

LE DÉVELOPPEMENT CROSS-PLATFORM : LA SOLUTION MIRACLE ?

Android, iOS, Windows Phone : trois technologies et autant de développements fondamentalement différents pour qui voudrait rendre son application compatible sur les trois plateformes.

Dès lors, la tentation du développement multiplateforme est grande... d'autant qu'elle propose souvent au développeur de faire l'économie de l'apprentissage d'un nouveau langage. La solution miracle existerait-elle donc ?

> Trois façons d'aborder le cross-platform

Plusieurs éditeurs proposent leur solution de développement smartphone cross-platform, mais toutes utilisent l'une des 3 stratégies suivantes :

- générer du code "natif" pour les différentes plateformes à partir d'un autre langage (ex : Titanium Appcelerator)
- faire fonctionner le code sur un runtime embarqué dans l'application (ex : Rhomobile)
- utiliser les capacités des composants "webviews" proposés par les différents SDK (ex : Phonegap)

La solution de la génération de code - qui paraît séduisante sur quelques cas d'utilisation simples - n'a jamais vraiment fonctionné dans l'histoire du développement : dès que les projets deviennent conséquents, cela se termine invariablement en soupe de code lourde et invraisemblable en termes de maintenance. Inévitablement, pour contourner les limites de ces solutions qui doivent toujours courir derrière les évolutions des différentes plateformes, la plupart des développeurs en viennent à faire eux-mêmes du code natif plus ou moins sauvage comme autant de sparadraps sur une jambe de bois. La méthode du runtime a pour elle le mérite d'une plus grande cohérence technique : le code est optimisé pour un runtime et les efforts du développeur peuvent se concentrer sur celui-ci. Cette approche qui est un peu celle de la JVM a néanmoins un défaut majeur pour l'univers du mobile : celui de rajouter une surcouche entre l'application et le système et donc de perdre notablement en performances à des niveaux souvent rédhibitoires pour un smartphone (manque de réactivité de l'interface, trop grande consommation de mémoire et d'énergie).

Reste la dernière possibilité, utiliser des webviews afin de coder en HTML/CSS/JS pour toutes les plateformes, qui loin d'être dépourvue de défauts, est une voie raisonnablement envisageable, au moins pour certains projets. Examinons donc le cas phonegap.

> L'architecture Phonegap / Cordova

Cordova (le projet open-source derrière Phonegap) fonctionne sur un principe simple : tout le code est réalisé en HTML5/CSS3/JS. Ce code est ensuite compilé au sein d'une webview (composant présent dans tous les SDK utilisant le moteur de rendu du navigateur web du smartphone) qui contient donc l'ensemble de l'application telle que le développeur l'a codée.

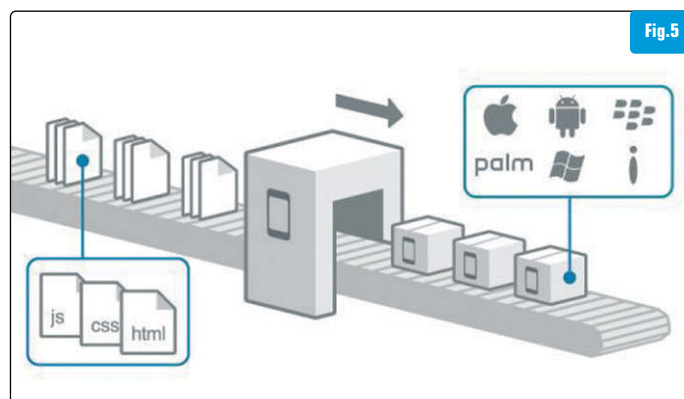
Reste que les webviews ont les limites du web et qu'elles ne permettent donc pas d'accéder aux fonctionnalités "natives" du téléphone (caméra, accéléromètre, boussole...). C'est là que Cordova prend tout son sens : ce beau projet propose en effet un ensemble de "bridges" rendant ces fonctionnalités natives disponibles au sein d'une simple API Javascript [Fig.5].

Du fait de cette architecture, le développement se passe donc en deux temps : tout d'abord la réalisation proprement dite de l'application au format web, puis sa compilation au sein des différents SDK ciblés (il faudra donc toujours Xcode et un Mac pour avoir une app iOS). Pour cette seconde étape, Adobe propose d'ailleurs un service, "Phonegap Build", afin d'effectuer pour vous et dans le cloud la compilation à destination des différentes plateformes visées.

> Développer avec Cordova

Repassons une fois encore par la case "Hello world" pour illustrer le développement sous Cordova. Pour cela, commençons par poser la base de notre contenu : du HTML5.

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<meta name="format-detection" content="telephone=no" />
<meta name="viewport" content="user-scalable=no, initial-scale=1, maximum-scale=1, minimum-scale=1, width=device-width, height=device-height, target-densitydpi=device-dpi" />
<title>Programmez !</title>
</head>
<body>
<div class="app">
<h1>Hello world demo</h1>
<input type="button" id="hello" value="hello">
</div>
<script type="text/javascript" src="cordova-2.1.0.js"></script>
<script type="text/javascript" src="js/index.js"></script>
```



Principe de fonctionnement de Cordova

```
<script type="text/javascript">app.initialize();</script>
</body>
</html>
```

Outre la structure HTML5 classique, on remarque l'insertion de 3 éléments de javascript : le chargement des bibliothèques Cordova, l'appel de notre code javascript de gestion et l'initialisation de l'application via la méthode `app.initialize()` de Cordova.

Attaquons-nous donc au code js en commençant par la variable `app` permettant l'initialisation de l'application :

```
var app = {
  //initialisation de l'application
  initialize: function() {
    this.bindEvents();
  },
  //initialisation des événements
  bindEvents: function() {
    document.addEventListener('deviceready', this.onDeviceReady, false);
  },
  var helloButton = document.getElementById('hello');
  helloButton.addEventListener('click', sayHello, false);
},
  //gestion de l'événement «application initialisée»
  onDeviceReady: function() {
    app.receiveEvent('deviceready');
  },
  // Sur l'événement receiveEvent, on crée un log (utile pour le debug)
  receiveEvent: function(id) {
    console.log('Received Event: ' + id);
  }
};
```

Il suffit ensuite d'écrire le code de la fonction `sayHello()` pour afficher un message au clic sur le bouton. Par exemple :

```
function sayHello(){
  alert("Hello world !");
}
```

Dernière étape : la compilation sous forme d'application. Celle-ci dépend de la plateforme que vous visez.

Il vous suffit de suivre les instructions de la documentation Phonegap pour Android ou pour iOS.

> Liens :

http://docs.phonegap.com/en/2.2.0/guide_getting-started_android_index.md.html et pour iOS :

http://docs.phonegap.com/en/2.2.0/guide_getting-started_ios_index.md.html

> Une solution viable... pour certains projets !

Cordova a quand même un certain nombre de défauts liés à sa conception. Premièrement, le fait de passer par une webview ajoute un niveau d'abstraction entre votre l'application et le système. Ainsi, même si les smartphones modernes optimisent plutôt bien l'interprétation des langages web, vous perdrez en performances.

Entendons nous bien : cela n'est pas forcément bloquant, mais si votre application doit comporter, par exemple, des animations ou des effets de transitions ultra-fluides, Cordova vous posera des problèmes.

Cordova ne peut pas non plus tirer parti de certains composants (graphiques, accès au hardware) natifs optimisés pour telle ou telle plateforme. Pour ce faire, il est nécessaire de passer par le système de plugins Cordova : mais vous perdez alors tout l'intérêt du multi-plateforme puisque vous devrez coder vos plugins pour chaque plateforme ciblée.

Le bon projet Phonegap est donc celui qui n'est pas trop exigeant en matière de performances et qui vous permettra de réaliser une application multiplateforme en ne nécessitant que peu de plugins.

Leçon 4 : *le développement Phonegap est une alternative viable au développement natif, peut-être même la seule. Mais c'est loin d'être toujours la meilleure option.*

LE BON CHOIX POUR VOUS ?

Terminons par un récapitulatif global de ces différentes possibilités pour le développeur... les notations données ici donnent une tendance

provenant à la fois des qualités fondamentales de chaque solution mais surtout de retours d'expérience concrets. L'évaluation de Windows Phone 8 est donc encore imprécise de ce point de vue.

Leçon 5 : *vous voulez gagner de l'argent en vendant sur un store ? Choisissez iOS. Vous voulez gagner de l'argent en développant pour d'autres smartphones ? Faites-vous embaucher :)*

	iOS	Android	Windows Phone	Phonegap
Langage standard	Obj-C	Java	C#, VB	HTML5/CSS3/JS
Langage « natif »		C / C++	C++	-
Fonctionnalités du SDK	+++	++++	+++	++
Productivité	+++	++	+++	++
Performances	++++	++	+++	+
Compatibilité tablettes	oui	oui	non	oui
Liberté de publication sur le store	+	++++	++	selon plate-forme
Chiffre d'affaires stores*	\$4 000 000 000	\$500 000 000	\$100 000 000	selon plate-forme
Parts de marché France**	51 %	39,00%	2 %	selon plate-forme
Types de terminaux cibles	10	> 1000	4	selon plate-forme

*estimations mondiales non officielles

**critère : utilisateurs web

Fig.6

Comparatif des technologies présentées

Jean-Baptiste Boisseau
Eutech SSI

La dictature des Publics Stores est-elle légitime ?

Lorsque nous avons un service à offrir sur le web, nous présentons celui-ci au travers de notre site. Nous élaborons une stratégie pour développer le trafic des visiteurs.

Nos clients s'inscrivent, sélectionnent un produit et l'achètent. Nous conservons leurs coordonnées et travaillons notre connaissance client pour les inciter à revenir et renouveler leur acte d'achat, le plus régulièrement possible.

Cette intimité client se construit progressivement, certes, sous le regard des moteurs de recherche et éventuellement via un appui publicitaire, mais finalement sans intermédiaires : ni ticket d'entrée, ni contrôle de notre relation client.

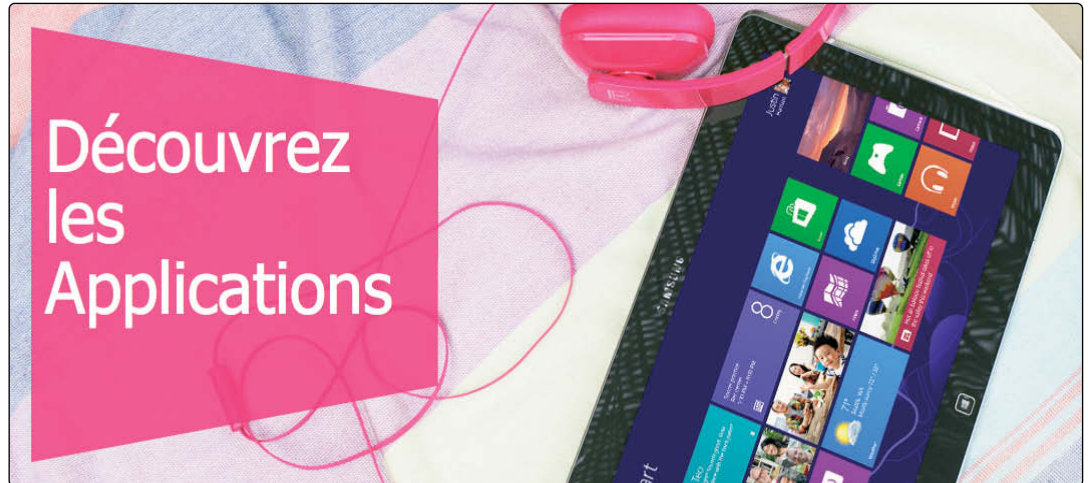
Notre site met en scène nos produits dans une vitrine dont nous gérons en fait tous les contours pour valoriser exclusivement nos offres.

Nos clients internautes deviennent mobinautes !

Comment faire pour exploiter leur mobilité et leur offrir de nouveaux contextes d'achat à la fois attractifs et efficaces ?

- Notre site peut devenir mobile et lisible sur tous types de devices grâce à du responsive design.
- Une webapp multi devices pourra, en plus, exploiter la fonctionnalité de géolocalisation.
- Une application hybride pourra, elle, tirer parti de la plupart des fonctionnalités spécifiques à chaque terminal : accéléromètre, boussole, caméra...
- Une application native offrira une expérience exclusive qui tirera parti de toutes les fonctionnalités de la plateforme mobile cible.

Plus nous cherchons à tirer parti des fonctions du terminal mobile, plus nous sommes amenés à publier des applications mobiles directement chargées sur les mobiles cibles.



Cette publication se fera sur l'un des grands « Public Stores ».

Ces espaces du commerce mobile appartiennent à des grands groupes qui les gèrent quasi exclusivement : Apple, Google, Microsoft, RIM... Il existe bien des « marchés alternatifs » pour Android, avec parmi eux, des « marchés légaux » et des « marchés noirs illégaux » mais ceux-ci restent anecdotiques.

Pour proposer nos applications il nous faut une approbation. Elle peut prendre entre 5 et 15 jours et donner lieu à une facturation, mais dans tous les cas, un pourcentage des ventes est reversé au « Public Store ».

Cependant, la forte valeur ajoutée de ces espaces de vente dédiés au canal mobile est intimement liée à leur fréquentation et au volume de téléchargements qu'ils génèrent. Ils permettent de capter et de recruter de nouveaux clients.

Le « Public Store » est-il pour autant une « tête de gondole » qui aide à vendre ?

Sur les « Publics Stores » finalement :

- Lorsque le mobinaute utilise comme clef de recherche les spécificités et avantages de nos produits, est-ce notre application qui apparaît en premier ?
- Sommes-nous suffisamment visibles parmi les milliers d'autres offres ?
- Notre capacité d'acquisition clients est-elle efficace ?
- Notre offre est-elle à l'abri de nos concurrents ?

- Acceptons-nous volontiers, éventuellement ou pas du tout, les règles imposées pour l'approbation d'une application : les délais, les coûts, la possibilité que notre application soit supprimée à distance sans préavis... ?

La solution serait de pouvoir :

- Distribuer une application directement sur la messagerie de nos clients.
- Optimiser le reporting de données marketing « mobiles analytiques »
- Gérer la publication des applications avec des fonctionnalités personnalisées et un niveau de segmentation client plus fin.

Ces fonctionnalités sont celles d'un « private store » mais celui-ci n'est autorisé par Apple que dans le cadre de publication « in house » c'est-à-dire sur la flotte de mobiles des collaborateurs en interne.

Les fonctions des terminaux et des OS jouent-elles un rôle commercial suffisamment fort pour justifier qu'elles soient monétisées de manière récurrente ?

Il existe, toutefois, entre les « publics stores » des différences de contraintes et de business model qu'il faut prendre en compte dès l'élaboration de la stratégie mobile.



Remy Buclin,
Corporate Business
Développeur Mobile &
Digital, Devoteam

Focus sur Windows Phone 8



Windows Phone 8 est la dernière plateforme de système d'exploitation mobile par Microsoft. La version initiale fut lancée fin 2010 avec une dizaine de modèles de téléphones et a reçu une mise à jour majeure appelée Mango, fin 2011. Nokia a commencé à proposer ses téléphones à ce moment-là. Depuis octobre 2012 Windows Phone 8 est disponible sur 7 téléphones des marques Nokia, HTC et Samsung, d'autres sont déjà annoncés.



Phone. Ces guidelines sont pensées afin de permettre à l'utilisateur de vivre une expérience uniforme d'une application à une autre. Dans ces guidelines on peut retrouver des choses simples comme le drag and drop ou plus complexes telles que certains types d'animations. Heureusement, si ces guidelines sont là pour simplifier la vie de l'utilisateur final, elles permettent aussi à tous les développeurs utilisant les contrôles du SDK de ne pas avoir à s'en préoccuper vu que ces contrôles respectent déjà toutes ces règles.

Bien entendu, en plus d'assurer la même expérience utilisateur d'une application à une autre, Microsoft étend cela au-delà du mobile puisque cette interface est semblable à celle de Windows 8, Windows RT, l'interface de la Xbox 360 ainsi que celle du baladeur Zune. De quoi peut-être donner l'idée à certains développeurs de créer une application sur ces divers supports [Fig.1].

> Uniformité

En plus de favoriser une expérience utilisateur semblable d'une application à une autre, Microsoft pousse la chose encore un peu plus loin en voulant la rendre semblable d'un Smartphone sous Windows Phone 8 à un autre. C'est dans ce but d'uniformisation que Microsoft exige de ses partenaires une configuration minimale pour les Smartphones destinés à accueillir Windows Phone 8. Ces spécificités requises sont on ne peut plus simples, mais assez précises en la matière : 4Go de mémoire de stockage minimum, un processeur dual-core de type Qualcomm Snapdragon S4, 1 Go de mémoire RAM (ou 512 pour les écrans WVGA), écran tactile multi-touch capacitif, caméra arrière avec autofocus et flash LED ou Xenon, GPS, accéléromètre, détecteur de proximité, capteur de luminosité, moteur de vibration, wifi 802.11 b/g, Bluetooth, gestion de DirectX, port Micro-USB 2.0.

> Live tiles

Les live tiles sont apparues avec la version 7.0 de Windows Phone. Elles correspondent aux tuiles présentes sur l'écran d'accueil du téléphone. Elles sont dites «live» car elles retransmettent en temps réel les informations dont l'utilisateur veut avoir connaissance sans même avoir à lancer une application. L'écran d'accueil est personnalisable par l'utilisateur, c'est lui qui définit la taille et l'emplacement de ces tuiles. Le développeur, de son côté, a la possibilité de mettre à jour les informations de la tuile de son application de manière locale ou distante sans que l'utilisateur ait à lancer l'application. Il est possible de modifier l'image de la tuile, le badge associé (ie : le nombre de notifications en attente) et le/les textes informatifs. Depuis la version 8 de Windows Phone, on peut facilement recréer le même comportement que les applications natives pour le design et l'animation des tuiles : contenus différents selon la taille de la tuile, animations cycliques de la tuile, comme l'application native Photos qui fait défiler des photos de l'album du téléphone à intervalle régulier.

A l'opposé de l'ancienne plateforme Windows Mobile, Windows Phone repart de zéro. Un nouveau design apparaît sous le nom de Metro mais s'appelle désormais Modern UI. L'information est mise en avant de façon imagée ou colorée avec le minimum de travail sur les formes (bordures, chromes). La typographie devient reine dans un monde de tuiles aux couleurs unies qui s'animent d'informations de tous les jours. L'authenticité de l'ergonomie se démarque profondément en laissant toujours la place à de nouvelles façons de réaliser graphiquement vos applications. Les applications panorama s'ouvrent comme une première de magazine de par leur dynamisme. Pour les professionnels, Windows Phone intègre les outils de tous les jours : Word, Excel, OneNote, PowerPoint et offre une connectivité vers SharePoint. Pour les passionnés de musique ou vidéo, le hub music+video offre une collection imposante pour un abonnement mensuel. Finalement, les développeurs d'applications retrouvent les outils qu'ils connaissent déjà bien : Visual Studio 2012 et Expression Blend.

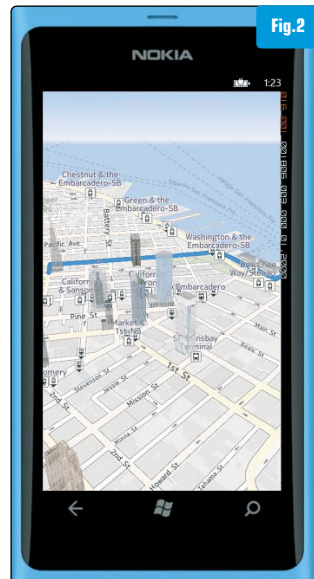
> Modern UI

Windows Phone 8 amène une nouvelle interface : Modern UI. En vérité, il s'agit plutôt d'une évolution de l'ancienne interface de Windows Phone 7. Une nouveauté importante de cette interface est sans doute la possibilité pour l'utilisateur de modifier la taille des tuiles, trois tailles étant disponibles. Cela s'ajoutant aux tuiles dynamiques et aux nouvelles couleurs d'interface, la possibilité de personnaliser son Windows phone augmente donc encore un peu plus. Mais au-delà de l'aspect utilisateur, Windows phone 8 apporte des guidelines à suivre pour tous les créateurs d'applications Windows

> Nokia Navtech Map Control

Sous Windows Phone 7, le seul moyen d'intégrer des cartes aux applications était d'utiliser le Control Bing Map du SDK. Sous Windows Phone 8, ce contrôle cède sa place au Nokia Navtech Map Control. Ce contrôle apporte toute une série de nouveautés par rapport à son prédécesseur, allant du paysage 3D jusqu'à l'intégration de certaines informations utiles aux piétons, l'info trafic...

D'un point de vue développement, le contrôle Nokia remplace complètement celui de Bing, ce qui implique seulement des méthodes en plus pour le développeur. Le contrôle Nokia apporte en plus des fonctionnalités déjà présentes dans l'application Nokia Drive comme les cartes de Jour et de Nuit, des cartes satellites et aériennes mais aussi des cartes indiquant le relief du terrain. Enfin le plus gros avantage qu'apporte ce nouveau contrôle est sans aucun doute la capacité de télécharger des cartes afin de les utiliser en mode offline [Fig.2].



> API vocale

La plateforme Windows Phone inclut un composant de reconnaissance et de synthèse vocale dans le téléphone. Arrivé avec la version 7 il se présentait sous le nom de TellMe et permettait aux utilisateurs de faire quelques opérations de manière vocale (appeler un contact, faire une recherche sur internet). Depuis la toute dernière version (8.0), le vocal sur Windows Phone a fait un bond en avant et rattrape, voire devance, les alternatives.

TellMe change de nom pour l'API Speech, et propose désormais la possibilité aux développeurs d'utiliser la reconnaissance et la synthèse vocale au sein de leur application. Hormis le mode dictée classique (la phrase de l'utilisateur est entièrement reconnue), l'API speech permet de gérer des « grammaires » (liste finie de mots) ce qui évite les problématiques récurrentes dans le domaine de la reconnaissance vocale. A savoir : les mots trop spécifiques qui ne sont pas présents ou mal reconnus par la plateforme de reconnaissance. Cette API permet aussi de faire de la synthèse vocale (le texte est lu par le téléphone) de manière avancée grâce à l'intégration de SSML (Speech Synthesis Markup Language Reference) comme mode de description du texte. Il est possible par exemple de définir la rapidité à laquelle doit être lue un texte.

La reconnaissance et la synthèse vocale fonctionnent dans différentes langues. Il est possible d'utiliser sur un même téléphone des langues différentes que ce soit pour la synthèse ou la reconnaissance vocale, à noter que toutes les langues ne sont pas installées par défaut. De plus, l'environnement Windows Phone permet au développeur d'ajouter des commandes vocales pour le démarrage de son application. De base, il est possible de lancer une application en disant « Démarrer application », mais la plateforme va plus loin en proposant aux développeurs d'ajouter des commandes vocales comme « Démarrer application et rechercher information », qui aura pour effet d'amener l'utilisateur directement à la fonctionnalité recherchée.

> Portefeuille

Nouvelle fonctionnalité arrivant avec Windows Phone 8, le Portefeuille. Il va simplifier la vie de l'utilisateur en permettant le regroupement des cartes de crédits, de fidélité et des coupons de réductions. Ce portefeuille électronique permet aussi de conserver l'historique des transactions. Petite ombre au tableau, cette fonctionnalité n'est pas compatible avec tous les modèles de téléphones. L'API propose aux développeurs d'interagir avec cette application, étendant idéalement les applications e-commerce [Fig.3].

> Vente d'applications

Au niveau de la monétisation d'application, Windows Phone 8 reprend les mêmes éléments de Windows Phone 7. On retrouve donc le moyen de publier une application en version gratuite ou payante. Une application payante peut être mise à disposition « à l'essai », laissant au créateur d'application le choix des fonctionnalités actives durant la période d'essai. La période d'essai pouvant être illimitée, cela permet de ne publier qu'une seule fois l'application si l'on souhaite la rendre disponible en version payante et gratuite.

Du point de vue des applications payantes, le Store Microsoft permet de vendre des applications entre 0.99\$ et 499.99\$. Les prix dans d'autres monnaies d'échange sont calculés en fonction de taux de changes fixes. 70% du prix de vente de l'application reviendra au développeur, les 30% restant allant à Microsoft.

> Store et publication

Afin de publier votre application, vous devez créer un compte sur le Windows Phone Store. Un abonnement annuel est nécessaire pour pouvoir publier et maintenir vos applications. Si vous êtes étudiant et que votre université adhère au programme Microsoft Dreamspark, alors vous aurez droit à un an d'abonnement gratuit. La publication est une étape critique pour votre application. Le Store impose des règles et des tests qui doivent être respectés. Parmi les règles les plus importantes figurent : demander la permission à l'utilisateur pour l'accès à la localisation ou au push, ne pas dépasser les limites d'utilisation mémoire, fournir des captures d'écran, fournir des icônes dans les tailles demandées. Une fois la publication effectuée, les équipes de Microsoft vont tester votre application afin de vérifier sa conformité. Tout problème entraînera l'envoi d'un rapport de tests stipulant les éléments bloquants. Entre autres, l'application ne doit pas « planter », l'interface doit rester réactive tout le temps, l'utilisateur doit toujours pouvoir recevoir un appel sans dérangement, le design doit être soigné... Ce processus rigoureux est en place pour éviter de proposer aux utilisateurs des applications instables ou ne répondant pas aux règles.



> Publicité

Afin de monétiser des applications gratuites, Microsoft propose dans le SDK un framework appelé le Ad Framework. Il permet d'intégrer facilement un contrôle d'affichage de publicité dans l'application. Une fois le contrôle intégré, il suffit de s'enregistrer sur le Microsoft Advertising Pub Center afin de déterminer le type de publicité affiché dans l'application. Bien entendu, rien n'empêche d'intégrer un autre type de contrôle de publicité.

> In-app purchase

En plus de leur contenu de base, certaines applications peuvent évoluer au fil du temps et permettre l'utilisation de nouvelles fonctionnalités. Les développeurs de ces applications peuvent faire payer ces fonctionnalités supplémentaires afin d'augmenter leurs revenus. Pour cela, Microsoft met à disposition via l'API le moyen d'acheter directement dans l'application de nouvelles fonctionnalités.

Chaque fonctionnalité aura un Token qui permettra de l'identifier dans l'application mais aussi dans le Windows Store. Ce Token est une chaîne de caractères permettant d'identifier le block de code correspondant à une fonctionnalité dans le code, ainsi que de développer plus rapidement l'interface graphique d'achat à l'intérieur de l'application. Bien sûr, le prix de ces fonctionnalités devra être déterminé dans la partie Advanced Features du processus d'enregistrement d'applications sur le Store.

> Enterprise Store

A côté du Windows Store pour les utilisateurs normaux, il existe un Windows store spécial pour les entreprises. Ce Store un peu particulier est disponible pour toute entreprise ayant un compte Microsoft Portail. Chaque entreprise dispose de son propre store où les applications doivent être signées et où il faut un compte entreprise afin de s'y connecter. Sur ce type de Store, les applications publiées ne passent pas les différentes étapes de certifications comme pour une application grand public, c'est l'entreprise qui est tenue comme responsable de l'application. Il est néanmoins toujours possible pour celle-ci d'utiliser le kit de test utilisé pour certifier les applications. L'avantage de ce type de store en interne, c'est qu'une fois connecté avec un compte de l'entreprise, toutes les applications nécessaires sont automatiquement installées et mises à jour. Bien entendu, rien n'empêche l'administrateur du Store de republier des applications grand public sur le Store interne de l'entreprise.

> Réseaux sociaux

Windows Phone permet à l'utilisateur de configurer ses profils sociaux d'une façon unifiée. Les réseaux supportés sont Facebook, Twitter et LinkedIn. Le premier avantage est l'intégration des flux sociaux dans les applications préinstallées. Le second avantage est de pouvoir publier sur ces réseaux depuis vos applications par l'usage d'une tâche de partage. Deux tâches sont disponibles : partager un statut et partager un lien.

```
ShareStatusTask shareStatusTask = new ShareStatusTask();
shareStatusTask.Status = «I'm developing a Windows Phone application!»;
shareStatusTask.Show();
```

Malheureusement, ces tâches de partage sont limitées en termes de fonctions disponibles. Si vous souhaitez effectuer des tâches de partage complexes depuis votre application, comme publier une

photo, il faudra le développer vous-même par code ou par l'utilisation d'une librairie tierce.

> Communication app-to-app

Windows Phone 8 intègre aussi un modèle de communication app-to-app afin de rendre possibles des scénarios comme démarrer une conversation Skype depuis une autre application. Pour activer la réception de communication pour votre application, il suffit de déclarer dans le manifest que vous souhaitez supporter la réception d'une URL dont vous déciderez de la structure. Le manifest est un document XML dans lequel on déclare les prérequis de l'application et ses connectivités avec le reste du système. Imaginons que l'adresse «myapp:user=1234» désigne un utilisateur de votre application.

```
<Protocol Name=»myapp» TaskID=»_default»
    NavUriFragment=»encodedLaunchUri=%s» />
```

Ce simple code permet de s'abonner à la réception d'un lien d'utilisateur Myapp et permet à une application de demander à la vôtre d'ouvrir l'élément désiré. Envoyer le code par SMS fonctionne : le système crée un lien ouvrant l'application. Scanner le code par NFC permet aussi le lancement de l'application. Côté code, vous devez créer une classe héritant de UriMapperBase dans laquelle vous pourrez lire le lien par vous-même et provoquer l'ouverture d'une page de votre application comme vous le souhaitez. Pour appeler une autre application dont vous connaissez l'adresse associée, utilisez la class Launcher avec la méthode LaunchUriAsync.

```
Windows.System.Launcher.LaunchUriAsync(new Uri(«myapp:user=1234»));
```

Si aucune application n'est installée pour recevoir l'URL, l'utilisateur sera invité à consulter la liste de toutes les applications supportant le protocole dans le store. Si plusieurs applications répondent à la même extension de fichier, l'utilisateur sera systématiquement invité à choisir l'application souhaitée. En conclusion, nous allons pouvoir développer des applications à forte capacité de partage d'informations (publications sur réseaux sociaux, transfert par proximité...). Tous les moyens sont bons pour ouvrir une application avec leur adresse : par SMS, NFC, email, un lien sur une page web.

> Associations fichiers

Le SDK fournit la capacité à une application de s'associer à des types de fichier afin de laisser à l'utilisateur le choix de l'application à démarrer pour ouvrir un fichier (depuis un email, internet...). Cet extrait de fichier manifest montre comment indiquer que votre application peut ouvrir les fichiers C# dont l'extension est .cs. Comme pour les opérations app-to-app, il suffit d'hériter une nouvelle classe de UriMapperBase pour écrire le comportement d'ouverture du fichier dans votre application.

```
<Extensions>
  <FileTypeAssociation Name=»C# code file» TaskID=»_default» Nav
    UriFragment=»fileToken=%s»>
    <Logos>
      <Logo Size=»small» IsRelative=»true»>Assets/file-csharp-
        33x33.png</Logo>
      <Logo Size=»medium» IsRelative=»true»>Assets/file-csharp-
        69x69.png</Logo>
      <Logo Size=»large» IsRelative=»true»>Assets/file-csharp-
```



```
176x176.png</Logo>
</Logos>
<SupportedFileTypes>
  <FileType ContentType=»text/plain»>.cs</FileType>
</SupportedFileTypes>
</FileTypeAssociation>
</Extensions>
```

Il ne reste plus qu'à gérer l'arrivée d'un fichier dans votre application et à l'ouvrir. Si plusieurs applications supportent le même type de fichier, l'utilisateur sera invité à choisir l'application de son choix. Si aucune application n'est installée, il sera invité à en trouver une sur le Store.

> Clavier

Une fonctionnalité assez intéressante présente sous Windows Phone 8 est la possibilité de changer le type de clavier visuel selon le type d'entrée à réaliser. Si l'option paraît sympathique à utiliser afin de faciliter la vie de ses utilisateurs, elle peut sembler plus dure à mettre en place pour le développeur. C'est sur ce point justement que l'on se trompe étant donné que Microsoft a rendu les choses très simples. Un exemple valant mieux qu'un long discours :

XAML :

```
<TextBox InputScope=»Number» Name=»txtPhoneNumber» />
```

C# :

```
InputScope scope = new InputScope();
InputScopeName name = new InputScopeName();
name.NameValue = InputScopeNameValue.Number;
scope.Names.Add(name);
txtPhoneNumber.InputScope = scope;
```

On voit donc par cet exemple que le changement du type de clavier se réalise en quelques lignes de code. De quoi sans doute donner plus d'idées aux développeurs voulant créer des applications simples et pratiques à l'utilisation [Fig.4].

Puissance de Microsoft

Un des plus gros avantages dont dispose Windows Phone 8 est sans aucun doute son appartenance à l'armada Microsoft. En effet, la firme de Redmond lui permet de profiter d'un grand nombre d'applications et de services déjà éprouvés. D'un point de vue utilisateur on peut compter sur la compatibilité entre les versions mobiles et PC de la suite Office, la liaison entre Zune et la bibliothèque de musiques et de vidéos ainsi qu'avec le service Xbox Music. Skydrive vient aussi ajouter son lot de bénéfices à cette interconnexion de périphériques. D'un point de vue développeur le SDK de Skydrive permettra d'utiliser le compte stockage de l'utilisateur que ce soit pour la lecture ou l'écriture de données. En partant de ce qui a été dit ci-dessus, il serait tout à fait normal qu'un développeur vienne à créer une application Windows Phone qui utiliserait Skydrive afin de stocker un fichier de configuration qui serait lui-même utilisé pour la version PC de cette même application. Cela permettrait donc un développement en parallèle de version mobile et PC de futures applications.

Visual Studio et Blend

Windows Phone 8 amène avec lui des nouveaux outils de développement, Visual Studio 2012 et une nouvelle version de Blend. Comme

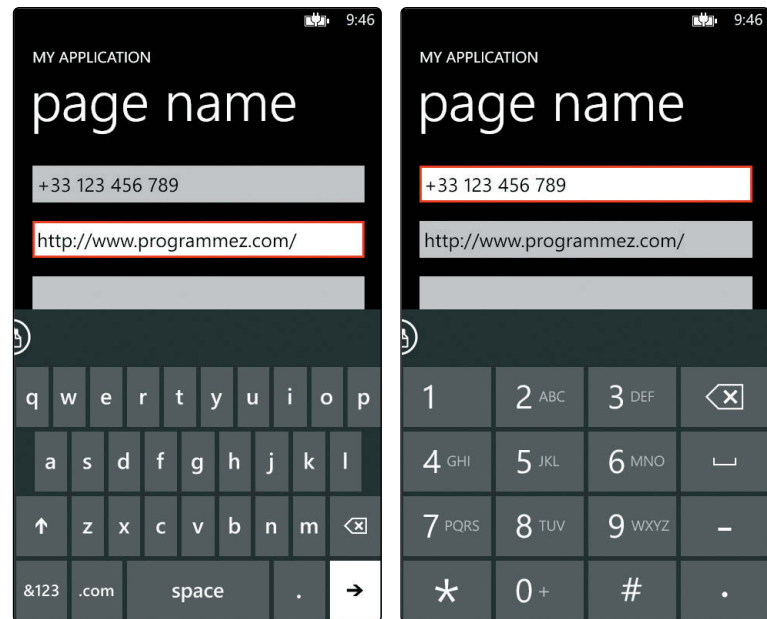


Fig.4

dans la version précédente, Blend est directement compris avec Visual Studio 2012, facilitant ainsi la vie des développeurs qui n'ont plus deux téléchargements et installations à réaliser. Ces deux outils très complémentaires permettent de développer des applications d'une meilleure richesse visuelle via Expression Blend qui autorise la création d'animations (ce qui est difficile avec Visual Studio). Cette nouvelle version de Blend apporte aussi des améliorations par rapport à son prédécesseur. On peut par exemple citer des outils de développement pour le CSS ainsi que l'IntelliSense pour le CSS. Visual Studio va vous permettre de développer vos applications en XAML+C# mais aussi en HTML ou XNA ou Direct3D (C++). Pour des raisons de sécurité, le SDK ne permet pas d'altérer le système comme on pourrait le faire sur d'autres plateformes. Vos applications sont exécutées dans un environnement isolé, empêchant l'écriture de code malicieux. Impossible donc d'accéder au système de fichier ou de créer votre propre clavier visuel. Windows Phone n'est donc pas un système profondément customisable mais plutôt un système robuste et sécuritaire.

Conclusion

En conclusion, Windows Phone 8 apporte toute une série d'améliorations et de fonctionnalités pour séduire les utilisateurs de Smartphones, mais aussi des outils de développement complets pour développeurs qui sauront sans doute apprécier l'attention que leur porte Microsoft avec ce nouvel OS mobile.

Ensuite, la qualité de ces nouveautés ainsi que l'énergie dépensée par les créateurs de cette nouvelle version montrent à quel point Microsoft se soucie d'étendre son public, que ce soit d'utilisateurs, en préservant l'expérience d'utilisation entre ses différentes plateformes, ou de développeurs, en essayant de leur simplifier la tâche au maximum. Enfin, Windows Phone 8 débarque sur le marché des Smartphones en se plaçant d'abord comme une évolution et non comme une simple mise à jour de Windows Phone 7.

Giovanni Clément, Antoine Sottiau,
Nicolas Tonneau, Mathias Herbaux
Wygwam



Développeur mobile : une carrière pleine d'avenir ?

Avec l'explosion du développement des applications pour smartphones et tablettes, le développeur doit-il envisager une carrière dans ces domaines, surtout avec la convergence entre les différents systèmes ? Il y a des opportunités à saisir. Mais il n'y aura pas de place pour tout le monde !

> Un profil exigeant

Qui est donc le développeur mobile (plus précisément pour smartphone et tablette) ? Nos différents entretiens avec des éditeurs et SSII dressent le profil suivant : c'est un développeur plutôt jeune (25-30 ans), avec souvent une sensibilité à l'ergonomie et au design (un point important qu'il ne faut pas négliger), il comprendra (un peu) mieux le fonctionnel et les usages (autre point sensible sur mobile), il a fait des développements professionnels, ou pour son loisir, il maîtrise au moins une plateforme mobile.



« Nous avons un grand nombre d'ingénieurs autour de 30 ans. Nos développeurs mobiles sont un peu plus jeunes. Certains développeurs se sont mis à la mobilité et il est important d'avoir des

développeurs qui connaissent les usages », précise **Emmanuel Gauthier** dirigeant d'AZE0.

« L'idéal est un développeur hybride connaissant iOS, Android et Windows 8 / Windows Phone 8. Souvent, le développeur a une expérience sur une plateforme. Nous avons aussi besoin de développeurs pointus sur un système, une technologie, par exemple, un développeur iOS maîtrisant le Core Anima-

tion », recadre **Christophe Claudel** (Président et fondateur d'Iteios). Le développeur ayant une vision globale de la mobilité avec une sensibilité au multiplateforme est un profil pas aussi fréquent qu'il n'y paraît, excepté sur la partie iOS / Android. Mais un des problèmes évoqués est que chaque plateforme a son propre modèle de développement...



L'autodidacte sera apprécié surtout s'il prouve ses qualités. La certification sur une ou plusieurs plateformes sera un plus sur votre C.V. mais pas l'ultime sésame pour se faire recruter.

> Un marché de l'emploi fluide

La demande de développement mobile explose depuis 2 ans. Les profils iOS / Android demeurent la référence. Windows Phone 8 est encore trop récent pour voir son réel impact sur le recrutement, mais des SSII spécialisées en technologies Microsoft recrutent et forment à celles-ci.

« Le marché est surtout difficile à appréhender ! C'est un peu l'effet que l'on a pu connaître à la fin des années 90 pour le web. Beaucoup d'intervenants disent avoir

des compétences, mais peu ont l'expertise nécessaire pour mener des projets de plus de 15 jours-hommes. Donc, lorsqu'on cherche les profils ayant cette expertise, le marché est clairement tendu », commente J-B Boisseau (Eutech SSII). « Nous trouvons beaucoup de développeurs. Mais le plus difficile est de faire le tri, nous voyons beaucoup de profils junior avec peu d'expérience », nuance Christophe Claudel. La candidature spontanée est aussi un bon moyen pour se faire recruter. La proportion de développeurs mobiles augmente rapidement : par exemple chez Eutech SSII, 50% des projets ont été liés à la mobilité en 2012. Chez Iteios, d'ici fin 2013, les développeurs mobiles représenteront un tiers des effectifs. Les SSII spécialisées et éditeurs mobiles continueront à recruter en 2013. Exemple d'annonces sur lesjeudis.com (au 12 décembre) :

- profils divers Android (avec parfois Android / iOS) : 56 offres d'emploi
- profils divers iOS : 40 offres (sans compter les offres spécifiques iPad)
- profils Windows Phone 8 : difficiles à trouver.

Salaire : très variable

« Les profils ayant déjà une expérience significative pourront bien se valoriser et de ce fait viser au-dessus des moyennes habituelles. La seule difficulté est que les employeurs/donneurs d'ordre ont encore du mal à faire la différence entre le développeur sachant un peu bricoler sur smartphone et un ingénieur de développement capable de prendre en charge un projet significatif », explique J-B Boisseau.

La fourchette de départ sera celle d'un développeur « classique », aux alentours de 28 / 30 K€. Les profils experts monteront facilement à 35, voire, 38/40 K€. Mais comme l'offre et la demande n'est pas en déséquilibre, nous ne constatons pas de pressions particulières sur les salaires.

François Tonic

Un exemple de recrutement : Genymobile

Genymobile, spécialisée en technologies mobiles et objets connectés, annonce rechercher 20 collaborateurs supplémentaires pour le 1er semestre 2013 (développeurs et ingénieurs systèmes Android, ainsi que des commerciaux, responsables communautaires, etc.). « Ce qui fait notre originalité est avant tout une offre unique sur le marché européen. En effet, nous sommes la seule société à proposer des solutions et services 100% Android. Grâce à nos équipes en place, nous développons des applications à forte valeur ajoutée. Ce succès, on le doit à nos collaborateurs, tous passionnés par leur métier. Genymobile est un laboratoire, où les idées se développent et les compétences se côtoient au quotidien afin d'offrir le meilleur à nos clients », explique Cédric Ravalec, le CEO de Genymobile.

Migrez vos applications vers ASP.NET 4.5 2^e partie

ASP.NET est un langage compilé et qui permet aux développeurs de réaliser des sites web dynamiques, des applications web et des Web Services tout en ayant accès aux fonctionnalités que peuvent offrir le Framework .NET (support du XML, les interactions avec la base de données, email, les expressions régulières...).

Commençons par créer notre première application Web API en ajoutant un nouveau projet ASP.NET MVC4 > Web API, avec le moteur de vue Razor [Fig.1]. Nous nous intéresserons dans ce premier exemple à 2 éléments du projet créé :

- Le répertoire Models
- Le répertoire Controllers

> Les modèles

Comme dans tout projet MVC, les modèles contiennent la logique métier. Ajoutons leur une nouvelle classe POCO Utilisateur :

```
namespace Programmez.Models
{
    public class User:
    {
        public int Id { get; set; }
        public string Login { get; set; }
        public string Password { get; set; }
    }
}
```

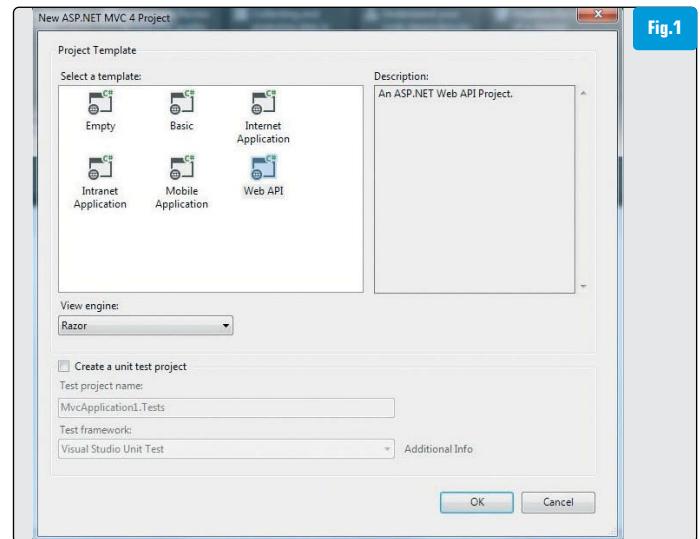
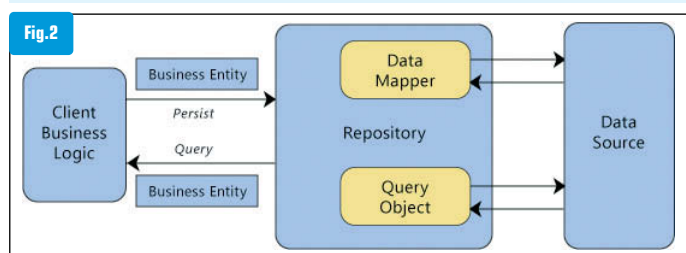
Nous voulons exposer une API d'administration permettant de manipuler ces utilisateurs. Pour ce faire, en nous basant sur le pattern Repository [Fig.2](<http://martinfowler.com/eaCatalog/repository.html>), nous allons ajouter une interface à notre modèle :

```
namespace Programmez.Models
{
    public interface IUserRepository
    {
        IEnumerable<User> GetAll();
        User Get(int id);
        User Add(Product item);
        void Remove(int id);
        bool Update(User item);
    }
}
```

Et ensuite une classe UserRepository :

```
namespace Programmez.Models
{

```



```
public class UserRepository : IUserRepository
{
    private List<User> users = new List<User>();
    private int _nextId = 1;

    public UserRepository ()
    {
        Add(new User { Login = «user1», Password = «pwd1» });
        Add(new User { Login = «user2», Password = «pwd2» });
        Add(new User { Login = «user3», Password = «pwd3» });
    }

    public IEnumerable<User> GetAll()
    {
        return users;
    }

    public User Get(int id)
    {
        return users.Find(p => p.Id == id);
    }

    public User Add(User item)
    {
        if (item == null)
        {
            throw new ArgumentNullException(«item»);
        }
        item.Id = _nextId++;
        users.Add(item);
    }
}
```



```

        return item;
    }

    public void Remove(int id)
    {
        users.RemoveAll(p => p.Id == id);
    }

    public bool Update(User item)
    {
        if (item == null)
        {
            throw new ArgumentNullException("item");
        }
        int index = users.FindIndex(p => p.Id == item.Id);
        if (index == -1)
        {
            return false;
        }
        users.RemoveAt(index);
        users.Add(item);
        return true;
    }
}

```

Nous voici donc prêts à manipuler nos données. C'est logiquement le contrôleur qui se charge de ces opérations.

> Le routage

Le protocole http expose 4 méthodes : GET, POST, PUT et DELETE. Chacune de ces méthodes sera associée à une ou plusieurs méthodes de nos contrôleurs de façon à interagir avec nos données. Par défaut, nous allons avoir les routes suivantes :

Action	HTTP method	Relative URI
Récupérer les users	GET	/api/users
Récupérer un user par son id	GET	/api/users/id
Récupérer les users dont le login correspond	GET	/api/users?login=loginPattern
Créer un user	POST	/api/users
Mettre à jour un user	PUT	/api/users/id
Supprimer un user	DELETE	/api/users/id

> Les contrôleurs

Clic droit sur le répertoire Controllers, ajouter un nouveau contrôleur, que nous allons nommer UsersController. Une classe UserController est générée dans le fichier Controllers/UserController.cs.

```

public class UsersController : ApiController
{
    static readonly IUserRepository repository = new UserRepository();
}

```

Notons tout d'abord qu'elle hérite d'ApiController. C'est cette classe qui définit notamment toutes les règles de configuration et de routage. En effet, à l'instar du DbContext d'EntityFramework, le fonctionnement par défaut d'ApiController est basé sur les conventions. Il est possible de personnaliser ce comportement en surchargeant cer-

taines méthodes. Notons ensuite que notre classe UsersController contient une variable repository de type IUserRepository. C'est donc le contrôleur qui fait office de passe-plat entre l'appel à l'API et le modèle. Enrichissons maintenant notre contrôleur :

```

public IEnumerable<User> GetAllUsers()
{
    return repository.GetAll();
}

public Product GetUser(int id)
{
    User item = repository.Get(id);
    if (item == null)
    {
        throw new HttpResponseException(HttpStatusCode.NotFound);
    }
    return item;
}

public HttpResponseMessage DeleteUser(int id)
{
    repository.Remove(id);
    return new HttpResponseMessage(HttpStatusCode.NoContent);
}

```

Par convention, Web API cherche une méthode commençant par le nom de la méthode http, et affine ensuite en fonction de la présente et le nom des paramètres. Ce comportement est débrayable à l'aide d'attributs. Notons ensuite que des classes HttpResponseMessage et HttpResponseException permettent de retourner des résultats http conformes.

Le résultat renvoyé par l'API est un message XML ou JSON par défaut, ou alors un format à définir, qui devra être géré côté client comme côté serveur.

Pour tester manuellement votre API, un navigateur suffira. En appelant une URL de l'API, un objet JSON ou XML sera renvoyé. Il est lisible dans un éditeur de texte. Pour consommer réellement le service, les possibilités sont larges ! Javascript JQuery, nouvelle classe .NET HttpClient, iOS, etc.

> Conclusion

Cet article a pour but de faire un rapide état des lieux sur comment et pourquoi migrer vos applications ASP.NET existantes vers ASP.NET 4.5. Nous sommes convaincus que cette migration vaut la peine d'être effectuée et elle peut d'ailleurs l'être de façon progressive. De forts gains en lisibilité et maintenabilité peuvent être espérés. Et enfin, MVC4 permet de bénéficier des apports de Web API, qui offre la possibilité d'exposer des services REST de façon simple et élégante.



Nicholas Suter
Architecte Logiciel chez Cellenza
Cellenza - Software Development Done Right
Blog : <http://www.cellenza.com/author/nsuter/>



David Tran
Consultant chez Cellenza
Cellenza - Software Development Done Right

Je débute avec le kit de démarrage ASP .net MVC

La technologie Open Source ASP.net MVC a le vent en poupe. Vous avez envie de vous lancer, n'hésitez pas ! Elle est excellente, c'est la meilleure pour développer des applications web pour navigateurs ou Smartphones. Débutons avec le kit de démarrage d'Innovacall.

D'apparence facile ce n'en est pas moins une technologie web qui nécessite une bonne maîtrise des technologies suivantes :

- HTML
- CSS
- JavaScript
- JQuery
- C# ou VB.net
- Razor ou ASP.net
- Entity Framework
- IIS 7.5
- SQL Server (ou compatible EF)

Les kits de démarrage Innovacall permettent d'éviter les adaptations nécessaires au contexte d'exploitation (Interne, externe ou Azure). Il ne s'agit donc pas d'une nouvelle technologie ou d'une version concurrente de MVC, au contraire, c'est plutôt un facilitateur destiné à des environnements plus «réalistes» que ceux des démos.

Sources et portail de démonstration (pour ASP.net MVC 4.0) :

<http://innovacallframework.codeplex.com>

<http://framework.innovacall.net>

Dans cet article, vous trouverez une description des principaux éléments qui constituent le kit de démarrage ainsi que la marche à suivre pour créer un nouveau projet.

> Contenu du kit de démarrage

- Framework
- Couches d'abstraction
- Dossiers du portail

Framework	<p><code>_Controllers_Base<T></code> : Contrôleur where T : <code>_Models_Base</code></p> <p>Premier niveau de factorisation des contrôleurs, classe générique qui hérite du premier niveau de factorisation des modèles. <code>_Models_Base</code> contient les membres qui alimentent la vue partagée : <code>_Layout</code>.</p> <p>Contient également de nombreux outils, extensions et helpers.</p> <p><i>Doit être partagé par tous les projets MVC du S.I. Les outils Peuvent-être utilisés par des projets non MVC.</i></p>
Business layer	<p>Contient le code métier, c'est-à-dire les fonctionnalités spécifiques à l'activité pour laquelle il est créé.</p> <p><i>Partagée par tous les projets MVC du SI qui en ont besoin.</i></p>
Common layer	<p>Couche dédiée à la factorisation des classes partagées par plusieurs couches. Contient les enums extraits de la base générés par un POCC T4 ; la méthode de rendu des vues partielles en chaîne de caractères permettant la conservation de la mécanique MVC dans les réponses Ajax.</p> <p><i>Partagée par tous les projets MVC du SI qui en ont besoin.</i></p>
Data layer	<p>Contient la classe <code>DataRepositories</code> (contexte de données qui permettent l'accès aux entités)</p> <p><i>Partagée par tous les projets MVC du SI qui en ont besoin (Générée automatiquement : ne doit pas être modifiée)</i></p>
Domain layer	<p>Contient les entités utilisées par le <code>DataRepository</code>.</p> <p><i>Chaque portail doté d'une EntityLayer et d'une DataLayer doit avoir une DomainLayer. (Générée automatiquement : ne doit pas être modifiée, sauf héritage d'interface à coder dans le POCC T4)</i></p>
Entity layer	<p>Contient l'EDMX d'Entity dont la stratégie de génération de code est désactivée.</p> <p><i>Spécifique à chaque portail (Seul l'EDMX peut être modifié, je suggère vivement une utilisation database-first)</i></p>

MetaData layer	<p>Contient les modèles partiels utilisés par les vues dont les membres sont automatiquement décorés des attributs nécessaires à la localisation et à la validation. Générée par POCC T4, les classes ont la même signature que les classes du domaine pour faciliter le mapping.</p> <p><i>Spécifique à chaque portail (Générée automatiquement : ne doit pas être modifiée)</i></p>
Resources layer	<p>Centralise l'ensemble des ressources pour la localisation.</p> <p><i>Spécifique à chaque portail (Peut-être partiellement générée automatiquement mais je ne fournis pas ce POCC T4)</i></p>
Security layer	<p>Roles & Membership provider s'appuyant sur Entity Framework. Il est possible de le personnaliser pour affiner l'isolation des données ligne par ligne.</p> <p><i>Spécifique à chaque portail (Générée automatiquement : ne doit pas être modifiée)</i></p>
Service layer	<p>Couche optionnelle destinée à centraliser les appels aux web services.</p> <p><i>Spécifique à chaque portail</i></p>
Content	CSS spécifiques aux composants clients séparés de ceux qui sont spécifiques aux vues.
Controllers	Contrôleurs MVC qui héritent de <code>Controller_Base<T></code> du Framework. Un contrôleur par vue.
Models	Structure hiérarchisée, le premier niveau contient le <code>ViewModel</code> qui hérite de <code>ModelBase</code> qui contient des <code>MetaModel</code> qui héritent des <code>MetaData</code> .
Scripts	Même principe que pour les CSS. Les scripts des composants clients sont séparés de ceux des vues.
Views	Vues MVC avec implémentation des <code>EditorTemplates</code> qui portent le nom des <code>MetaModels</code>

> Fichiers de configuration

AppSettings.config	Contient les paramètres de l'application (peut être complété par vos paramètres)
ConnexionString.config	Contient la ou les chaînes de connexion
JQueryCss.config	Contient la liste des fichiers javascript et css qui seront inclus dans toutes les pages
mailSettings.config	Contient le paramétrage pour l'émission des e-mails
web.config	Web.config classique

> Fondamentaux

Ce framework est plus strict que le MVC original. Il impose les règles suivantes :

- A chaque vue correspond un contrôleur et un seul
- Chaque contrôleur est spécialisé pour le `ViewModel`
- Le Contrôleur doit impérativement hériter de `_Controller_Base<T>` du framework
- Le `ViewModel` doit impérativement hériter de `_Model_Base` du framework
- Les données fonctionnelles du `ViewModel` sont encapsulées dans des `MetaModels` (classes qui héritent de `MetaData` générées à partir du modèle Entity Framework)

> Les modèles

Les modèles sont de simples classes destinées au transport des données entre le client et le serveur. Leurs membres sont décorés d'annotations nécessaires au nommage et à la validation des champs. Les modèles inclus sont donc plus structurés que les classes utilisées avec MVC original : les modèles du kit ont été « pré-factorisés » pour la « mécanique » de fonctionnement de la page :

4 Framework	_model_base	Membres permettant d'alimenter la vue _layout.cshtml
3 CommonLayer	_model_base	Membres et méthodes communes aux couches et au portail
2 Portal	_model_base	Membres et méthodes communes aux contrôleurs du portail
1 Portal	(View)Model	Membres et méthodes nécessaires à l'initialisation de la vue

Initialisation de la vue _Layout.cshtml

```
@using INNOVACALL.Framework.Common
@using InnovaApp.Layers.ResourcesLayer.Properties
@{
    InnovaApp.Layers.CommonLayer.Models._Models_Base layoutModel =
    (InnovaApp.Layers.CommonLayer.Models._Models_Base)Model;
}
```

Et pour la partie « données fonctionnelles »

1 Portal	MetaModel	Membres permettant d'alimenter les choix et les collections
2 MetaDataLayer	MetaData	Membres décorés des attributs MVC

Fig.1

Initialisation des données du modèle dans l'exemple Sample_BasicForm_MyProfile

```
MembershipUser currentUser = Membership.GetUser();
Contact dbContact = _dataContext.Contact_Rep.Query(x => x.FK_Inno
AppUser == (Guid)currentUser.ProviderUserKey).FirstOrDefault();
if (dbContact == null)
{
    ContentData.Contact = new Models.MetaModels.ContactManagement.
    Contact_MetaModel();
}
else
```

Diagramme des classes modèle de l'exemple Sample_BasicForm_MyProfile du portail de démo.

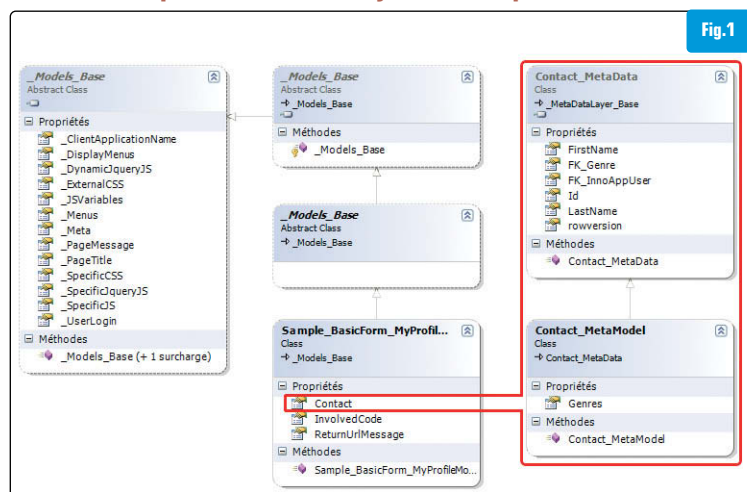


Fig.1

La variable correspondant au ViewModel dans le contrôleur s'appelle toujours ContentData.

```
{
    ContentData.Contact = ObjectMapperManager.DefaultInstance.Get
Mapper<Contact, Contact_MetaModel>().Map(dbContact);
}
List<SelectListItem> Genres = _dataContext.Genre_Rep.GetAll().
OrderBy(x => x.SortIndex).Select(x => new SelectListItem() { Value =
x.Id.ToString(), Text = x.Name, Selected = (x.Id == ContentData.
Contact.FK_Genre) }).ToList();
Genres.Insert(0, new SelectListItem() { Value = «-1», Text = string.
Empty, Selected = (dbContact == null) });
ContentData.Contact.Genres = Genres;
```

> Les contrôleurs

Comme pour les modèles, la logique suit celle du MVC original et ils sont également « pré-factorisés » pour que les méthodes d'alimentation des membres factorisés des modèles soient en correspondance avec ces derniers.

Le comportement original est garanti par héritage :

Pour le contrôleur de base du Framework

```
public abstract class Controllers_Base<T> : Controller where T :
INNOVACALL.Framework.Common.Models._Models_Base
```

Pour un contrôleur du portail

```
public class Sample_BasicForm_MyProfileController : _Controllers
_Base<Sample_BasicForm_MyProfileModel>
```

Fig.2

Cela permet d'une part de forcer tous les contrôleurs d'hériter de la Classe Controller native MVC pour garantir un comportement similaire à un contrôleur standard et d'autre part de spécialiser chaque contrôleur pour une classe qui hérite de _Modele_Base créant ainsi un parallèle entre la structure factorisée du modèle et celle du contrôleur de sorte qu'à chaque héritage, on trouve les traitements correspondant aux membres du modèle du même niveau.

Initialisation des variables qui alimentent la méthode Init-View qui instancie le modèle

```
protected override void Initialize(System.Web.Routing.Request
Context requestContext)
{
    base.eMethodBase = «InnovaApp.Portals.InnovaAppFrontOfficePortal.
Controllers.Sample_BasicForm_MyProfileController»;
    base.specificCssCollection = new string[] { «_FormCommon.css»,
```

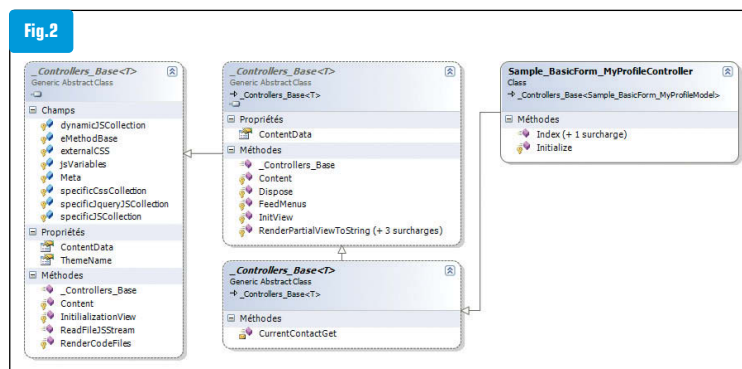


Diagramme des classes Controller de l'exemple Sample_BasicForm_MyProfile



```

«Views/Sample_BasicForm_MyProfile.css», «ClientComponents/Code
Formatting/csharp.css», «ClientComponents/KendoUI/kendo.common
.min.css», «ClientComponents/KendoUI/kendo.blueopal.min.css» };
base.specificJSCollection = null;
base.dynamicJSCollection = new string[] { «/ScriptsG/inoSample
_BasicForm_MyProfileDynamic.js» };
base.specificJqueryJSCollection = new string[] { «ClientComponents
/KendoUI/kendo.web.min.js», «InnovacallJS/Sample_BasicForm_My
Profile/Sample_BasicForm_MyProfile.js» };
base.externalCSS = null;
base.Meta.Add(enMeta.Description, Resources.View_Sample_Basic
Form_MyProfile_Meta_Description);
base.Meta.Add(enMeta.Keywords, Resources.View_Sample_BasicForm
_MyProfile_OrderWithDetails_Meta_Keywords);
base.Meta.Add(enMeta.Subject, Resources.View_Sample_BasicForm
_MyProfile_OrderWithDetails_Meta_Subject);
base.Initialize(requestContext);
}

```

Préparation du premier affichage de la vue (puisque toutes les vues sont Ajax)

```

[HttpGet, Authorize, CacheManager(enCacheMode.NoCache)]
public ActionResult Index()
{
    {
        using (DataRepositories _dataContext = new DataRepositories())
        {
            base.InitView(
                Resources.View_Sample_SingleControl_DropDownList_PageTitle,
                Resources.View_Sample_SingleControl_DropDownList_PageMessage,
                specificCssCollection,
                specificJSCollection,
                dynamicJSCollection,
                specificJqueryJSCollection,
                jsVariables,
                externalCSS,
                Meta,
                _dataContext,
                True
            );
        }
    }
}

```

Chargement des données dans le modèle via le ContentData instancié par InitView ci-dessus

```

if (Request.QueryString[«ReturnUrl»] != null)
{
    ContentData.ReturnUrlMessage = Resources.View_Sample_
BasicForm_MyProfile_PageMessage_Redirected;
}
MembershipUser currentUser = Membership.GetUser();
Contact dbContact = _dataContext.Contact_Rep.Query(x => x.FK_
InnoAppUser == (Guid)currentUser.ProviderUserKey).FirstOrDefault();

if (dbContact == null)
{
    ContentData.Contact = new Models.MetaModels.Contact
Management.Contact_MetaModel();
}
else

```

```

{
    ContentData.Contact = ObjectMapperManager.DefaultInstance.
GetMapper<Contact, Contact_MetaModel>().Map(dbContact);
}

List<SelectListItem> Genres = _dataContext.Genre_Rep.Get
All().OrderBy(x => x.SortIndex).Select(x => new SelectListItem()
{ Value = x.Id.ToString(), Text = x.Name, Selected = (x.Id ==
ContentData.Contact.FK_Genre) }).ToList();
Genres.Insert(0, new SelectListItem() { Value = «-1»,
Text = string.Empty, Selected = (dbContact == null) });
ContentData.Contact.Genres = Genres;
}
return View(ContentData);
}
}

```

Traitement transactionnel du post Ajax des données du formulaire Ajax de la vue

```

[HttpPost, Authorize, CacheManager(enCacheMode.NoCache)]
public JsonResult Index(Sample_BasicForm_MyProfileModel model)
{
    string eMethod = eMethodBase + «Index[Post]»;
    string returnUrl = Request.QueryString[«ReturnUrl»];
    string scriptToExecute = string.Empty;
    object eParameters = new { @modelContactId = model.Contact.Id,
@modelContactFK_Genre = model.Contact.FK_Genre, @modelContact
FirstName = model.Contact.FirstName, @modelContactLastName = model.
Contact.LastName };
    JsonResult toReturn = new JsonResult { JsonRequestBehavior = Json
RequestBehavior.AllowGet };
    using (TransactionScope TS = new TransactionScope())
    {
        try
        {
            {
                using (DataRepositories _dataContext = new DataRepositories())
                {
                    Guid currentUserId = (Guid)Membership.GetUser().Provider
UserKey;
                    Contact dbContact = _dataContext.Contact_Rep.Query(x => x.FK_
_InnoAppUser == currentUserId).FirstOrDefault();
                    if (dbContact == null)
                    {
                        dbContact = new Contact();
                        dbContact.FK_InnoAppUser = currentUserId;
                    }
                    dbContact.FK_Genre = model.Contact.FK_Genre;
                    dbContact.FirstName = model.Contact.FirstName;
                    dbContact.LastName = model.Contact.LastName;
                    _dataContext.Contact_Rep.AddUpdate(dbContact);
                    _dataContext.Save();
                    TS.Complete();
                }
            }
            if (!string.IsNullOrEmpty(returnUrl))
            {
                scriptToExecute = «$.fn.NavigateTo('« + returnUrl + «')»;
            }
            toReturn.Data = new { @success = true, @scriptToExecute = script

```

```
ToExecute, @divToUpdate = string.Empty };
    return toReturn;
}
catch (Exception ex)
{
    TS.Dispose();
    throw ex;
}
}
```

> Les vues

Seule évolution par rapport aux vues standards : inutile d'utiliser la génération des liens vers les fichiers javascript et css puisque tout est pris en charge dans la vue maître.

La combinaison des modèles structurés et des *EditorTemplates* permet de créer des vues complexes simplifiant les mises à jour partielles et le post final de toute la hiérarchie.

Vue de l'exemple *Sample_BasicForm_MyProfile*

```
@using InnovaApp.Layers.ResourcesLayer.Properties
@model InnovaApp.Portals.MVC4Portal.Models.Sample_BasicForm_MyProfileModel
@section featured {
    <section class="featured">
        <div class="content-wrapper">
            <hgroup class="title">
                <h1>@Model._PageTitle</h1>
                <h2>@Model._PageMessage</h2>
            </hgroup>
            @if (!string.IsNullOrEmpty(Model.ReturnUrlMessage))
            {
                <p class="yellowText">@Model.ReturnUrlMessage</p>
            }
            <p class="message-info">
                @Resources.View_Sample_BasicForm_MyProfile_Help.
            </p>
        </div>
    </section>
}
@using (Ajax.BeginForm(new AjaxOptions() { HttpMethod = "Post",
OnSuccess = "$.fn.MainAjaxSuccess", OnFailure = "$.fn.MainAjaxFailure" })))
{
    <section>
        <fieldset class="centerMargin25 ui-widget ui-widget-content ui-corner-all padding15">
            <legend>@Resources.View_Sample_BasicForm_MyProfile_FieldsetCaption</legend>
            <div>
                @Html.EditorFor(m => m.Contact)
            </div>
            <input type="submit" value="@Resources.View_Sample_BasicForm_MyProfile_Submit" id="cmdSubmit" />
        </fieldset>
    </section>
}
```

> EditorTemplates et vues partielles

L'EditorTemplate doit porter le même nom que la classe (MetaModel) qui l'alimente. Aucun d'entre eux n'a pour suffixe MetaData pour une meilleure homogénéité. En pratique, même si un MetaData se suffit à lui-même on crée quand même un MetaModel qui hérite du MetaData pour améliorer l'évolutivité.

EditorTemplate de la fiche contact

```
@model InnovaApp.Portals.MVC4Portal.Models.MetaModels.ContactManagement.Contact_MetaModel
<ol>
    <li>
        @Html.LabelFor(m => m.FK_Genre)
        @Html.DropDownListFor(m => m.FK_Genre, Model.Genres)
        @Html.ValidationMessageFor(m => m.FK_Genre)
    </li>
    <li>
        @Html.LabelFor(m => m.FirstName)
        @Html.TextBoxFor(m => m.FirstName, new { @class = "k-input" })
        @Html.ValidationMessageFor(m => m.FirstName)
    </li>
    <li>
        @Html.LabelFor(m => m.LastName)
        @Html.TextBoxFor(m => m.LastName, new { @class = "k-input" })
        @Html.ValidationMessageFor(m => m.LastName)
    </li>
</ol>
```

(Notez l'utilisation de *Html.LabelFor* et *Html.ValidationMessageFor* conformément au Standard MVC4)

> Les formulaires Ajax

Les vues avec formulaire utilisent toujours le formulaire Ajax

```
@using (Ajax.BeginForm(new AjaxOptions() { HttpMethod = "Post",
OnSuccess = "$.fn.MainAjaxSuccess", OnFailure = "$.fn.MainAjaxFailure" })))
{
    . . .
}
```

Modification de l'attribut action du formulaire pour diriger l'appel Ajax vers la bonne méthode

```
$.fn.AddDetailRow = function (evt) {
    . . .
    $(«form»).attr(«action», «/AdvancedSamples_OrderWithDetails/
AddRow»);
}
```

Ils sont toujours paramétrés pour utiliser les méthodes JQuery génériques *\$.fn.MainAjaxSuccess* et *\$.fn.MainAjaxFailure*. Seule *\$.fn.MainAjaxSuccess* doit être adaptée.

```
$.fn.MainAjaxSuccess = function (a, b, c) {
    if (a.success) {
        if (a.divToUpdate != null) {
            if (a.htmlContent != null) {
                $(a.divToUpdate).html(a.htmlContent);
            } else {
```

```
$(a.divToUpdate).html(«»);
}
if (a.scriptToExecute != null) {
    eval(a.scriptToExecute);
}
}
}
```

Si **Success=false** le traitement a échoué sur le plan fonctionnel alors que s'il s'agit d'une erreur Ajax elle sera traitée par **\$.fn.MainAjax-Failure**. **divToUpdate** est l'élément dont le contenu HTML sera remplacé par **htmlContent**. Enfin **scriptToExecute** permet d'effectuer des traitements après la mise à jour, par exemple un binding d'Event à une fonction JQuery.

> Le portail de démonstration

C'est le Starter Kit, Il suffit donc de supprimer les éléments inutiles dans la table des menus pour désactiver les exemples. Voici ses fonctionnalités.

LogOn	Presque identique à la version du modèle de projet Internet original
LogOff	Seul le contrôleur existe (pas de vue). Lorsque le contrôleur est invoqué, il clôture la session et redirige la réponse vers la page d'accueil.
Register & Activation	Saisie des informations minimales pour la création d'un compte : E-Mail / Login / Password. Le compte est créé et un mail contenant un lien vers une landing page d'activation de compte est envoyé à l'utilisateur.
ChangePassword	Accessible en cliquant sur le nom de l'utilisateur connecté.
Text box Ajax	Démo d'un petit callback Ajax sans passer par un formulaire.
Drop Down List Ajax	Présentation de la mécanique de remplissage d'une liste déroulante à partir de la base de données, et du traitement Ajax du changement d'option.
Mon profil	Formulaire simple avec persistance en base
Gestion d'adresses	Liste des adresses / Saisie d'adresse
Prise de commande	Formulaire complet de prise de commande avec choix des adresses et du mode de livraison et gestion Ajax des détails.

> Création d'un nouveau projet Préparation

- Télécharger la dernière version sur Codeplex : <http://innovacallframework.codeplex.com>
- Exécuter le script de génération de la base de données (SQL Server 2008 R2).
- S'assurer que la version du fichier kendo.web.min.js soit bien la dernière version de kendo ui

- Modifier les fichiers de configuration pour qu'ils correspondent à votre environnement
- Compilez, lancez le projet

Modification de la base de données

Vous pourrez modifier la base de données toutefois, si vous ne souhaitez pas conserver les fonctionnalités natives d'identification

Table	Fonctionnalité	Dépendance
SEC.InnoApplication	Role & Membership	Security Layer; LogOn, LogOff, Register, BasicSample_MyProfile, Advanced Sample Address Manager, Advanced Sample Order / Details
SEC.InnoAppUser	Role & Membership	Security Layer; LogOn, LogOff, Register, BasicSample_MyProfile, Advanced Sample Address Manager, Advanced Sample Order / Details
SEC.InnoProfile	Role & Membership	Security Layer
SEC.InnoRole	Role & Membership	Security Layer
SEC.InnoUsersInRoles	Role & Membership	Security Layer
TLS.LayoutMenu	Menu principal	Toutes les pages du portail de démonstration à l'exception de Login, et register

Une fois vos modifications faites, il vous suffit de régénérer les POCO T4.

> Conclusion

Dans cet article, nous avons survolé les principales caractéristiques de ce kit de démarrage pour ASP.net MVC 4 et les quelques opérations nécessaires à la création d'un nouveau projet s'appuyant sur cette boîte à outils. ASP.net MVC 4 est encore jeune, le kit de démarrage Open Source n'est publié que depuis quelques semaines. Plusieurs outils complémentaires sont prévus comme une extension pour Visual Studio qui automatisera plusieurs tâches qui sont encore répétitives, un enrichissement de la documentation et des vidéos de formation.

La prochaine fois, je présenterai un tutoriel plus complet sur la création d'une vue et de ses dépendances permettant d'éditer son propre contenu à l'aide d'un éditeur WYSIWYG en introduction de MyPrettyCMS, notre petit CMS Open Source.

J'espère sincèrement que tout ou partie de ces sources vous seront utiles.

Laurent Jordi
PDG Innovacal



ABONNEMENT
PDF

30 € par an

soit 2,73 € le numéro
www.programmez.com

Abonnement INTÉGRAL
Pour un supplément de 10 € an
accès illimité aux archives

Cette option est réservée aux abonnés pour 1 an au magazine, quel que soit le type d'abonnement (Standard, Numérique, Etudiant). Le prix de leur abonnement normal est majoré de 10 € (prix identique pour toutes zones géographiques). Pendant la durée de leur abonnement, ils ont ainsi accès, en supplément, à tous les anciens numéros et articles/ dossiers parus.

Prise en main de C++ AMP

Que vous soyez développeur C++ ou bien même C# sans connaissance sur la programmation GPU, cet article vous permettra de comprendre à la fois les concepts inhérents à la programmation sur GPU et leurs implémentations respectives via la librairie C++ AMP. Si au contraire, vous êtes déjà familier avec les technologies comme CUDA ou OpenCL, cet article vous permettra d'apprécier l'expressivité de C++ AMP, à la fois simple et efficace.

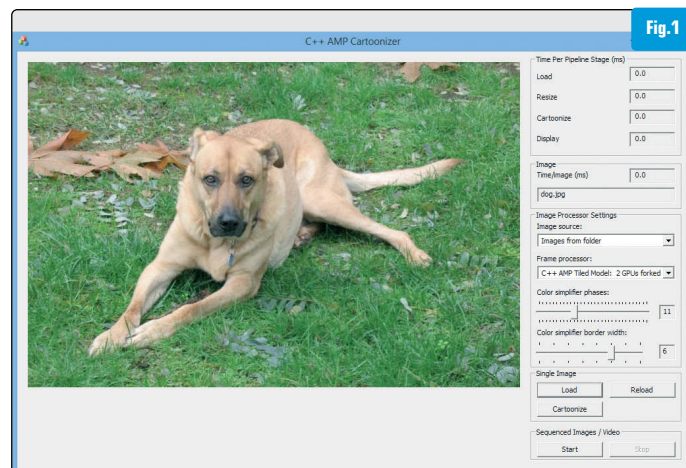
Remarque : Toutes les démonstrations sont réalisées avec Visual Studio 2012 sous Windows 8 x64, le tout exécuté sur un portable Lenovo W520 contenant un processeur Intel i7260QM cadencé à 2,6 GHz et une carte graphique NVIDIA Quadro M2000.

Si vous êtes complètement novice en programmation parallèle sur GPU, je vous propose une démonstration provenant du premier ouvrage traitant de C++ AMP de Kate Gregory et Ade Miller: <http://www.gregcons.com/cppamp/>, dont les codes sources sont téléchargeables : <http://ampbook.codeplex.com/releases/view/93158>.

La démonstration repose sur une des trois études de cas du livre. Cette démonstration se nomme **Cartoonizer**. Elle est parfois utilisée par Microsoft pour démontrer C++ AMP. La motivation de cette application est de modifier les couleurs des contours d'une ou plusieurs photos sur le disque ou prises par la Webcam de votre ordinateur en une image similaire à un dessin animé.

Au lancement de l'application, si vous chargez la première image par défaut, vous obtiendrez l'image ci-contre. L'application regroupe sous plusieurs thèmes des informations relatives au traitement des images. Si vous cliquez sur le bouton « Start », toutes les images du dossier par défaut seront traitées, mais nous n'utiliserons pas ce mode de traitement dans cette démonstration. Nous utiliserons essentiellement les boutons regroupés sous le nom « Single Image », afin de traiter une image à la fois [Fig.1].

L'objectif de cette première démonstration est de vous sensibiliser à la puissance GPU, et non d'étudier le code de cette application. Dans la liste « **Frame Processor** » vous sélectionnez la technologie de parallélisation et dans la liste « **Image source** » choisissez le mode récupération des images. Vous devez cliquer sur « **Load** » pour charger la première image par défaut. Pour notre premier test, nous choisissons le mode « **Image from folder** » et « **CPU Single Core** » pour la liste « **Frame processor** ». Puis vous cliquez sur le bouton « **Cartoonize** » [Fig.2].

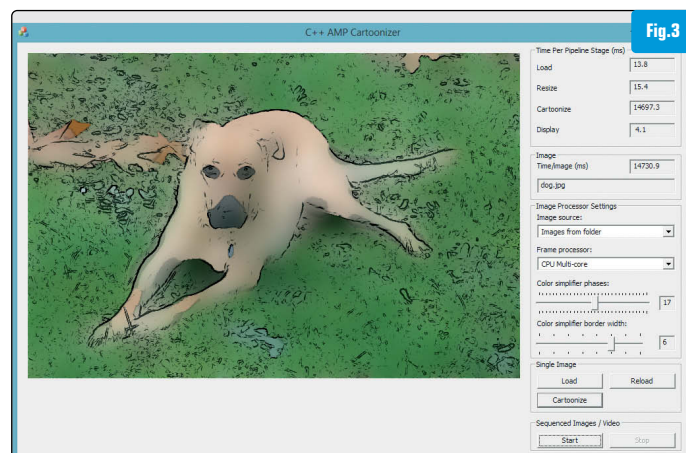
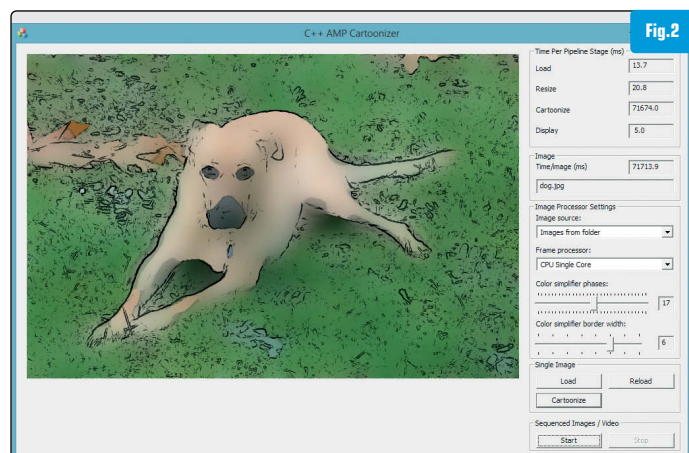


Vous devriez obtenir un résultat similaire et dans mon cas, ce premier traitement utilisant un seul CPU a pris environ **72 secondes**.

Passons au second traitement en choisissant cette fois, « **CPU Multi-core** » dans la liste « **Frame processor** ». Puis chargez à nouveau la première image et lancez le traitement en appuyant sur « **Cartoonize** » [Fig.3]. Ce traitement repose en interne sur la librairie parallèle C++ **PPL** disponible depuis Visual Studio 2010 [Fig.4].

En affichant le « **Task Manager** », on peut observer la charge CPU au moment du traitement. Le traitement utilisant tous les cœurs du processeur a pris **15 secondes** environ. Il nous offre une bien meilleure performance, ce qui n'a rien d'étonnant.

Cette fois, vous allez sélectionner dans la liste « **Frame processors** » l'item « **C++ AMP Simple Model : WRAP** ». Cet item correspond au mode de repli de C++ AMP dans le cas où votre machine ne contient aucune carte graphique compatible avec son implémentation interne (nous reviendrons sur ce point un peu plus loin) [Fig.5]. Cette fois, le « **Task Manager** » affiche un profil CPU bien plus



maigre que la version précédente [Fig.6].

Noter que le mode **WARP** est vraiment performant, car il n'a pris que 3 secondes, même s'il n'utilise pas de technique GPU. C'est donc une excellente alternative au traitement GPU, nous en reparlerons. Cette fois, vous allez sélectionner dans la liste « **Frame processors** » l'item « **C++ AMP Simple Model : Single GPU** ». Rechargez l'image et lancez le traitement [Fig.7].

Le premier traitement utilisant une technologie GPU n'a pris que **351 millisecondes**, ce qui est bien plus rapide que tous les autres traitements. Voici un récapitulatif de tous les résultats de cette première démonstration [Fig.8].

Cette première démonstration vous permet de comprendre la puissance du calcul avec C++ AMP. Je vous propose maintenant de comprendre ce qu'est C++ AMP, en expliquant les motivations qui ont permis à Microsoft de se lancer dans ce projet.

> Contexte

Lorsque Microsoft ajoute une nouvelle librairie à Visual Studio, ce n'est pas sans plusieurs motivations parfaitement réfléchies et cette nouvelle librairie, n'échappe pas la règle. Certains lecteurs pourraient être surpris par ce choix, car des technologies comme CUDA et OpenCL disponibles depuis plusieurs années ont rencontré un franc succès auprès des développeurs spécialisés GPU. Pour qu'il n'y ait aucun doute sur les motivations de C++ AMP, je vous propose de passer en revue tous les éléments qui ont poussé Microsoft à se lancer sur un sujet aussi intimiste que la programmation GPU.

> Rendre la programmation GPU accessible à tous

La grande majorité des développeurs produit des programmes à destination d'un ou plusieurs CPU, mais finalement elle produit du code ciblant la technologie CPU. Les CPU sont par nature modélisés pour réaliser des traitements génériques sans considération spécifique. Ils sont parfaitement adaptés à la majorité des besoins réclamés par les programmes actuels. Par exemple des applica-

tions riches, des applications web et serveurs applicatifs, sont généralement orientés données, mais ne réclament pas des traitements massivement parallèles. Par nature, les cartes graphiques sont destinées à calculer des informations graphiques pour afficher de très nombreux pixels sur un ou plusieurs écrans. Cependant, certains développeurs graphiques dans les années 2005 ont trouvé astucieux d'utiliser les cartes graphiques plutôt que le CPU pour accélérer des calculs sur des volumes de données très importants. Le calcul sur carte graphique est pertinent lorsque votre problème repose sur un immense volume de données à traiter.

Les cartes GPU possèdent des architectures plus simples que les CPU, mais ont surtout un nombre impressionnant de threads que n'ont pas les CPU. Aujourd'hui encore, les architectures matérielles des cartes graphiques sont construites pour obtenir des performances d'affichage exceptionnelles. Le succès du calcul sur GPU de ces dernières années a donné lieu à des cartes professionnelles sans sortie vidéo, complètement spécialisées dans la programmation massivement parallèle sur GPU.

Le problème est qu'aujourd'hui, peu de gens possèdent des compétences pour programmer les GPU. Ce type de programmation reste l'affaire de quelques spécialistes que l'on trouve généralement dans les services R&D, de nombreuses industries réclamant des calculs très coûteux, mais aussi dans le domaine des jeux où on utilise souvent la puissance des GPU. En d'autres termes, les développeurs généralistes ne connaissent pas la programmation GPU, car ils sont rarement face à des traitements coûteux, mais surtout parce que la programmation GPU est une technologie de niche compliquée à mettre œuvre. Le développeur généraliste souhaite rarement s'investir dans une technologie intimiste souvent exprimée en langage C. Il est accompagné d'un écosystème relativement pauvre, même si parfois il se trouve face à des cas où des portions de code seraient bien plus performantes en programmation GPU, car il n'existe pas de solution grand public pour programmer les GPU aujourd'hui. Ce dernier point est très important, car c'est la première motivation qui a poussé Microsoft à lancer le projet C++ AMP.

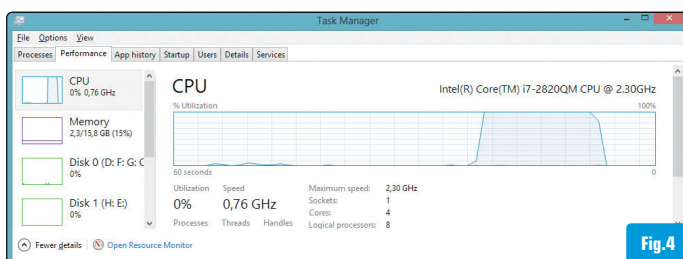


Fig.4

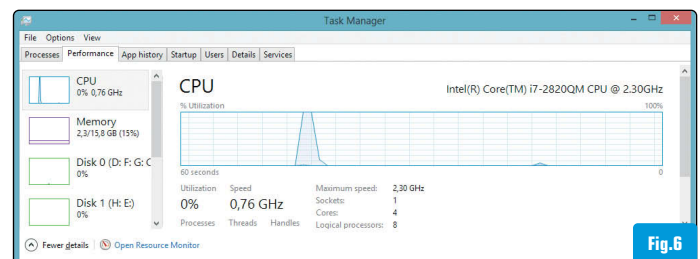


Fig.6

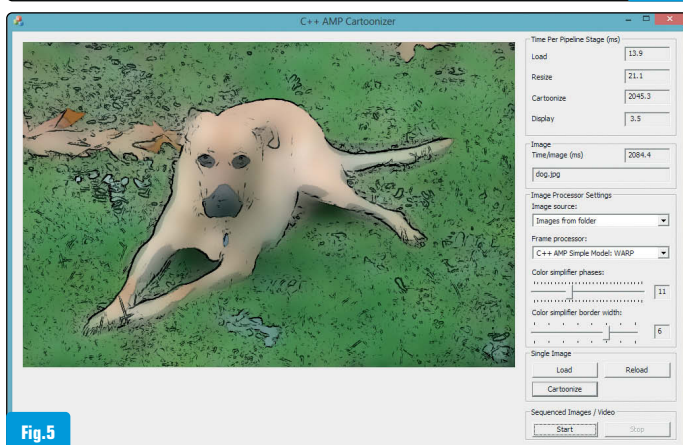


Fig.5

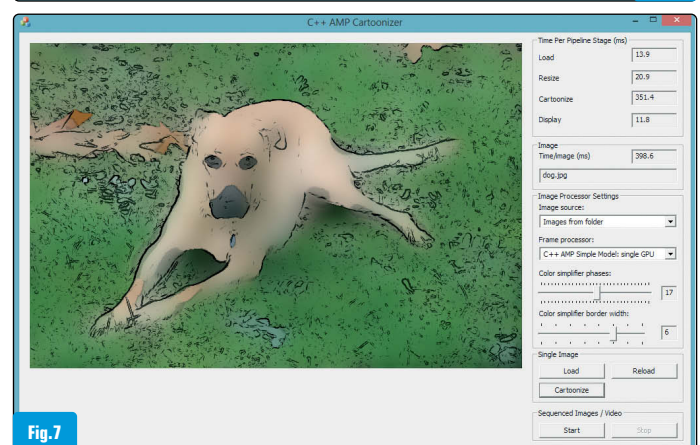


Fig.7

> Rendre agnostique votre code C++ AMP

Aujourd'hui, les constructeurs de cartes graphiques sont à la croisée des chemins en termes d'architecture matérielle. On assiste à des évolutions importantes entre les GPU et les CPU. C'est un secteur en pleine évolution où la compétition est rude et touche une poignée de constructeurs pour un marché immense. Pour les développeurs C++ AMP, les ingénieurs Microsoft ont pris en compte ce paramètre et affirment que tous les codes C++ AMP ne souffriront pas des futures évolutions matérielles. En réalité, votre code C++ AMP est dès à présent capable de s'exécuter sur de nombreux matériels sans aucune modification. Par nature C++ AMP n'est pas rattaché à un matériel spécifique (pas de code spécifique aux cartes graphiques nVidia, AMD ou même Intel). Autrement dit, votre investissement sera préservé. Ce dernier point est très important, car c'est la seconde motivation qui a poussé Microsoft à lancer le projet C++ AMP.

> Vous offrir un large choix de plateforme d'exécution à terme

Votre investissement sur C++ AMP ne se limite pas à une seule plateforme : de Windows Azure à Windows Phone, de Windows Desktop à Windows RT, de Windows Server à Windows Embedded, Windows HPC Server à Xbox, toutes ces plateformes exécuteront à terme votre code C++ AMP. Mais ce n'est pas tout, dans un futur proche, vous pourrez exécuter du code C++ AMP sur des plateformes non Microsoft. En effet, Microsoft a publié en février 2012, une spécification ouverte sur le standard C++ AMP (<http://download.microsoft.com/download/4/0/E/40EA02D8-23A7-4BD2-AD3A-0BFFB640F28/CppAMPLanguageAndProgrammingModel.pdf>), permettant aux fabricants de compilateurs d'implémenter C++ AMP sur des plateformes complémentaires à celles de Microsoft. Aujourd'hui, AMD a déjà annoncé qu'il produira une version de C++ AMP au-dessus d'Open CL. L'initiative de Microsoft à travers la spécification ouverte de C++ AMP, a pour but d'encourager tous les fabricants de compilateurs d'implémenter la librairie C++ AMP sur diverses plateformes. Vous l'aurez compris, le champ des possibles de C++ AMP sera à terme immense et vous n'avez pas à vous soucier de la plateforme d'exécution qui exécutera votre code, ce point est très important, car c'est à la fois la troisième et dernière motivation qui a incité Microsoft à lancer le projet C++ AMP. Arrivé à ce stade, vous avez pris connaissance des motivations de Microsoft à produire la librairie C++ AMP, il est temps de décrire ce qu'est C++ AMP.

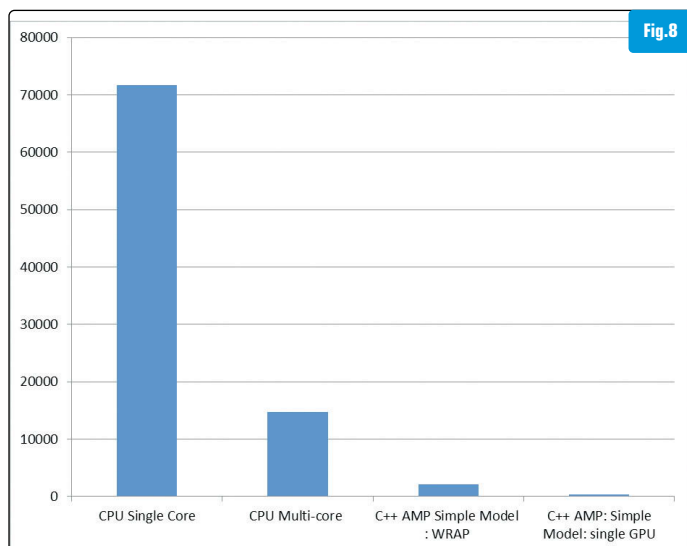


Fig.8

> Qu'est ce que C++ AMP ?

C++ AMP fait partie du compilateur C++. Si vous utilisez déjà le compilateur C++ de Visual Studio 2012, vous avez C++ AMP. Vous n'avez besoin de rien d'autre. Pour déployer votre application C++ AMP, vous n'avez aucun prérequis supplémentaire, le redistribuable Visual C++ contient la librairie C++ AMP. En tant que partie intégrante à Visual Studio, la librairie C++ AMP est parfaitement intégrée à Visual Studio à la fois sur le plan du Debugging, du Profiling et de l'IntelliSense. Avec C++ AMP, vous pouvez réutiliser vos applications et vos connaissances C++, car la librairie a été pensée pour justement les préserver. On peut définir C++ AMP comme une librairie C++ exposant un petit jeu d'API (compatible STL) sachant gérer des données multidimensionnelles afin de faciliter leur parallélisation. La courbe d'apprentissage de C++ AMP est donc faible, car le jeu d'API est réduit et repose sur le standard C++ 11. Si vous connaissez la librairie STL, alors vous connaissez une bonne partie de C++ AMP. L'implémentation Microsoft de C++ repose sur l'API Direct3D de DirectX 11. Ce choix d'implémentation est plutôt une bonne chose, car DirectX est une librairie mature et donc stable. Ce choix apporte une abstraction pour supporter de nombreux matériels comme ceux de constructeurs nVidia, AMD, Intel, ARM ... Cependant, si votre programme ne détecte pas de matériel compatible DirectX11, C++ AMP se tourne vers une solution de repli, WRAP, que nous avons utilisé dans la première démonstration, exploitant alors les multi cœurs disponibles et la vectorisation parallèle via les instructions SSE (AVX n'est pas supporté pour l'instant). Même si Microsoft utilise DirectX pour implémenter C++ AMP, les API DirectX ne sont pas visibles des API C++ AMP. C'est pour cette raison que Microsoft a pu publier une spécification ouverte décrivant le standard C++ AMP sans aucune adhérence avec DirectX. Pour Microsoft, C++ AMP se doit d'être performant, productif et portable pour tous les développeurs, ce n'est pas un produit de niche. Il n'est pas utile d'être un spécialiste en programmation GPU pour utiliser C++ AMP. Ainsi, un développeur C++ généraliste disposant de Visual C++ 2012, peut parfaitement utiliser cette technologie sans contrainte technique majeure : pas de nouvelles compétences, pas de librairies à ajouter pour la mise au point ou l'analyse de performance. La mise en production ne réclame rien de plus que le redistribuable C++. Et pour couronner le tout, votre code devrait tourner sur des plateformes complémentaires à celle de Microsoft dans un avenir proche. Arrivé à ce stade, nous avons beaucoup décrit C++ AMP, sans jamais l'illustrer avec du code, il est temps de corriger le tir.

> Hello world

Dans ce premier exemple, l'objectif est d'introduire les bases de C++ AMP progressivement. Si vous cherchez un intérêt vis-à-vis de la parallélisation : il n'y en a pas ! En revanche, vous apprendrez l'essentiel des éléments pour exploiter C++ AMP correctement. Pour vous convaincre de la simplicité de C++ AMP, démarrez Visual Studio 2012 et dans le Template C++, choisissez de créer une application, un projet vide que vous pouvez appeler « HelloWorld ». Puis sur le projet, ajoutez (clique droit) un nouvel élément source.cpp. Puis vous pouvez saisir ce code ou télécharger l'exemple.

```
#include <iostream>

void do_it(int* v, int size, int to_add)
{
```




```

for(int i = 0; i < size; i++)
{
    v[i] += to_add;
}

for(unsigned int i = 0; i < size; i++)
    std::cout << static_cast<char>(v[i]);
}

int main()
{
    int v[12] = {'G', 'd', 'k', 'k', 'n', 31, 'v', 'n', 'q', 'k', 'c', '-'};
    int to_add = 1;

    do_it(v, sizeof(v)/sizeof(int), to_add);

    std::cin.get();
}

```

Si vous exécutez ce code, vous obtiendrez le fameux « Hello World. ». Le code se compose d'un point d'entrée; La fonction **main** et d'une méthode **do_it** qui reçoit le tableau d'entier initialisé dans la fonction **main** et un entier **to_add** valorisé à un. La méthode **do_it** contient deux boucles.

La première calcule l'ajout de l'entier **to_add** sur tous les éléments du tableau, la seconde affiche tous les éléments une fois altérés. Naturellement, la seconde boucle ne fait qu'afficher et ne nous concerne pas sur plan de la parallélisation. Pour introduire C++ AMP, ajouter le fichier header « **amp.h** » et l'espace de nom « **concurrency** ».

```

#include <iostream>
#include <amp.h>
using namespace concurrency;

```

Dans la méthode **do_it**, nous allons remplacer la première boucle par la méthode **parallel_for_each**. Cette méthode existe aussi dans l'offre parallèle CPU (**Parallel Pattern Library** ou PPL), ce n'est pas cette version que nous allons utiliser, mais celle du fichier « **amp.h** ». Le premier paramètre réclame la taille de tableau à itérer, puis une expression lambda C++ 11.

```

void do_it(int* v, int size, int to_add)
{
    parallel_for_each(size, [](int i)
    {
        v[i] += to_add;
    });

    for(unsigned int i = 0; i < size; i++)
        std::cout << static_cast<char>(v[i]);
}

```

La méthode **parallel_for_each**, n'accepte pas une taille sous la forme d'un entier, mais d'un type permettant de traiter des données multidimensionnelles via le type template **extent<N>**. Dans notre exemple, nous n'avons qu'une seule dimension, mais le prochain exemple disposera de deux dimensions.

```

void do_it(int* v, int size, int to_add)
{
    extent<1> e(size);

    parallel_for_each(e, [](int i)
    {
        v[i] += to_add;
    });

    for(unsigned int i = 0; i < size; i++)
        std::cout << static_cast<char>(v[i]);
}

```

À ce stade, notre méthode est presque correcte, mais les données de notre tableau ne peuvent toujours pas être chargées dans le GPU. Nous allons utiliser un nouveau type template C++ AMP : **array_view<T,N>**. C'est un « wrapper » sur les données, ici notre pointeur d'entiers, permettant de les charger à la demande. Le type **array_view** est naturellement capable de wrapper des conteneurs STL. De plus, le type du paramètre de l'expression lambda doit être remplacé par le type : **index<N>**. C'est une sorte d'indice multidimensionnel, car il arrive souvent sur des projets plus réalistes que les données se représentent de manière multidimensionnelle.

```

void do_it(int* v, int size, int to_add)
{
    extent<1> e(size);
    array_view<int> av(1, v);
    parallel_for_each(e, [](index<1> i)
    {
        av[i[0]] += to_add;
    });

    for(unsigned int i = 0; i < size; i++)
        std::cout << static_cast<char>(v[i]);
}

```

> Nouveau mot clef C++ : restrict

Notre code n'est pas encore correct, car le corps de l'expression lambda constitue le code dit « noyau », c'est-à-dire, celui qui est chargé dans le GPU. Le code « noyau » ne peut pas exécuter toutes les syntaxes supportées par le C++. Pour que le compilateur puisse vérifier ces restrictions, nous devons ajouter un nouveau mot clef ajouté au C++ par C++ AMP : **restrict[amp]**. Nous remplaçons le tableau **v** par l'instance **av**, dans la seconde boucle.

```

void do_it(int* v, int size, int to_add)
{
    extent<1> e(size);
    array_view<int> av(size, v);

    parallel_for_each(av.extent, [](index<1> i) restrict(amp)
    {
        av[i[0]] += to_add;
    });

    for(unsigned int i = 0; i < size; i++)
        std::cout << static_cast<char>(av[i]);
}

```

Le principe de restriction peut s'appliquer sur une expression lambda (comme dans notre exemple) ou bien sur une méthode. Noter que le mot clé **restrict** modifie la signature de la méthode ou de l'expression lambda, ce qui signifie que nous pourrions ajouter par exemple une méthode nommée **foo()** marquée par le mot clef **restrict(amp)** sans soucis.

```
#include <iostream>
#include <amp.h>
using namespace concurrency;

void foo() restrict(amp) {}

void do_it(int* v, int size, int to_add)
{
    array_view<int> av(size, v);

    parallel_for_each(av.extent, [=](index<1> idx) restrict(amp)
    {
        foo();
        av[idx] += to_add;
    });

    for(unsigned int i = 0; i < size; i++)
        std::cout << static_cast<char>(av[i]);
}
```

À ce stade de l'article, nous pouvons prendre le temps d'expliquer le déroulement de notre calcul. La variable `idx` est renseignée automatiquement par le moteur d'exécution C++ AMP, car elle représente le thread GPU qui exécutera le code « noyau » (le corps de l'expression lambda). Chaque itération de l'expression est donc calculée indépendamment et en parallèle [Fig.9]. Pour l'instant l'instruction **restrict** ne supporte que les deux identifiants : **amp** et/ou **cpu**. Son rôle est d'informer le compilateur pour qu'il applique des restrictions au langage (restrictions spécifiques, optimisations, génération de code). Pour l'instant, le mot clef **restrict(amp)** interdit les éléments suivants :

- récursions
- mot clef volatile
- fonctions virtuelles
- pointeurs de fonctions
- pointeurs vers des fonctions membres
- pointeurs dans des structures
- pointeurs de pointeurs
- champs de bits
- goto ou labels
- throw, try, catch
- Variables globales ou statiques
- dynamic_cast ou typeid
- déclarations `__asm__`
- varargs
- types non supportés: char, short, long double

Si le code d'une méthode est à la fois compatible **amp** et **cpu**, vous pouvez marquer celle-ci des deux identifiants.

```
void foo() restrict(amp, cpu) {}

void do_it(int* v, int size, int to_add)
{
```

```
array_view<int> av(size, v);

parallel_for_each(av.extent, [=](index<1> idx) restrict(amp)
{
    foo();
    av[idx] += to_add;
});

for(unsigned int i = 0; i < size; i++)
    std::cout << static_cast<char>(av[i]);
}
```

Par défaut, tout votre code est compatible à la mode **cpu**, il est donc inutile de marquer votre code **restrict(cpu)**, excepté si vous souhaitez surcharger une méthode portant une signature équivalente, mais marquée avec **restrict(amp)**

```
void foo() restrict(amp) {}
void foo() {}

void do_it(int* v, int size, int to_add)
{
    array_view<int> av(size, v);

    parallel_for_each(av.extent, [=](index<1> idx) restrict(amp)
    {
        foo();
        av[idx] += to_add;
    });

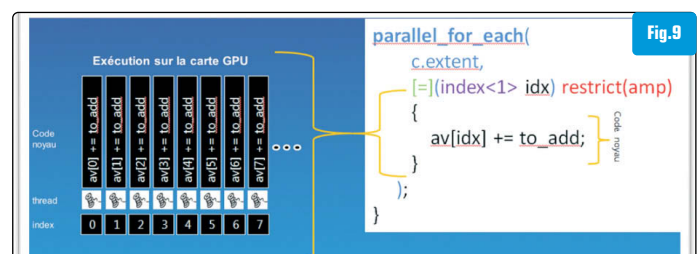
    for(unsigned int i = 0; i < size; i++)
        std::cout << static_cast<char>(av[i]);
}
```

Nous pouvons remanier un peu ce code, afin d'obtenir un code un peu plus élégant. Nous pouvons noter que le code final est à peine plus grand que le code d'origine.

```
void do_it(int* v, int size, int to_add)
{
    array_view<int> av(size, v);

    parallel_for_each(av.extent, [=](index<1> idx) restrict(amp)
    {
        av[idx] += to_add;
    });

    for(unsigned int i = 0; i < size; i++)
        std::cout << static_cast<char>(av[i]);
}
```





> Comment gérer l'exécution asynchrone du code GPU

Le code de la fonction `do_it` peut vous sembler parfaitement séquentiel, alors que toute exécution sur un GPU est par définition asynchrone. Alors qui est responsable de la synchronisation ? À la fin de la première boucle, nous avons fini d'écrire dans la variable `av`, dans la seconde boucle on parcourt l'`array_view` à nouveau, mais dans ce cas, qui synchronise la première lecture de la première boucle vis-à-vis de la lecture dans la seconde ? La classe `array_view` assure la synchronisation et évite tous problèmes asynchrones. Prenons un autre exemple, où cette fois nous ajoutons une seconde boucle d'écriture.

```
void do_it(int* v, int size, int to_add)
{
    array_view<int> av(size, v);

    parallel_for_each(av.extent, [=](index<1> idx) restrict(amp)
    {
        av[idx] += to_add;
    });

    parallel_for_each(av.extent, [=](index<1> idx) restrict(amp)
    {
        av[idx] += 2;
    });

    for(int i = 0; i < size; i++)
        std::cout << static_cast<char>(v[i]);
}
```

À la fin de la première boucle, la seconde boucle reprend à zéro l'écriture dans l'`array_view`, et une fois encore l'`array_view` va se synchroniser de manière à entamer la première écriture une fois que toutes écritures de boucle précédente seront terminées. Dans l'exemple suivant, nous avons déplacé le code d'affichage dans la fonction `main`. La classe `array_view` est instanciée sur la pile de la méthode `do_it`, ainsi à la sortie de méthode, le destructeur de l'`array_view` est appelé. Ce destructeur contient un code de synchronisation, permettant de « flusher » dans l'objet wrappé, ici, le pointeur d'entier, tous les éléments traités dans la méthode noyau.

```
void do_it(int* v, int size, int to_add)
{
    array_view<int> av(size, v);

    parallel_for_each(av.extent, [=](index<1> idx) restrict(amp)
    {
        av[idx] += to_add;
    });
}

int main()
{
    int v[12] = {'G', 'd', 'k', 'k', 'n', 31, 'v', 'n', 'q', 'k', 'c', '-'};
    int to_add = 1;
    int size = sizeof(v)/sizeof(int);
    do_it(v, size, to_add);
}
```

```
for(int i = 0; i < size; i++)
    std::cout << static_cast<char>(v[i]);
std::cin.get();
}
```

Cependant, il est aussi possible d'appeler sur l'instance de l'`array_view`, la méthode `synchronize` qui assure la synchronisation des données.

```
void do_it(int* v, int size, int to_add)
{
    array_view<int> av(size, v);

    parallel_for_each(av.extent, [=](index<1> idx) restrict(amp)
    {
        av[idx] += to_add;
    });

    av.synchronize();
}
```

Remarque : Il n'est pas nécessaire d'appeler la méthode `synchronize` car comme nous l'avons dit précédemment la classe `array_view` contient dans son destructeur le code pour synchroniser les données à la sortie de la méthode `do_it`. Cependant, l'usage de cette méthode vous assure que votre code peut rester correct si vous ajoutez du code à la fin de la méthode exploitant les données du tampon d'entier `v`.

> Accelerator et accelerator_view

Pour s'exécuter, un programme C++ AMP se doit d'utiliser un accélérateur (rappelez-vous la démonstration de l'introduction), c'est-à-dire une cible d'exécution où périphérique d'exécution. Actuellement dans la version 1.0 de C++ AMP, nous disposons des accélérateurs suivants :

- `accelerator::default_accelerator`
- `accelerator::direct3d_warp`
- `accelerator::direct3d_ref`
- `accelerator::cpu_accelerator`

Un accélérateur est généralement attaché à une carte graphique, mais il peut arriver que l'accélérateur cible votre CPU car aucune carte graphique compatible **DirectX 11** n'a été trouvée au démarrage de l'application. Dans ce cas, c'est l'accélérateur **WARP** ou **Microsoft Basic Render Driver** qui est utilisé. Rappelez-vous dans la première démonstration, l'exécution avec le mode WRAP nous avons obtenu des performances supérieures au code parallèle CPU. Pour comprendre cet exploit, sachez que l'accélérateur WARP utilise à la fois le parallélisme multi cœur et les instructions Intel SSE qui permettent de réaliser du SIMD pour Single Instruction Multiple Data, afin d'exécuter efficacement la parallélisation de vos données sur CPU. Sur le plan de l'implémentation cet accélérateur repose sur le service **WARP software rasterizer**, fourni par le moteur d'exécution DirectX 11.

Par défaut c'est l'accélérateur `accelerator::default_accelerator`, que le moteur d'exécution C++ AMP sélectionne, en choisissant la solution la plus performante vis-à-vis de votre configuration matérielle. Si vous souhaitez faire de la mise au point du code noyau, vous ne pourrez pas utiliser l'accélérateur le plus performant, mais l'opposé, le plus lent : `accelerator::direct3d_ref`. Le nom REF vient de «Reference Rasterizer» qui est une implémentation logicielle que le SDK DirectX a toujours fournie. C'est effectivement, le plus lent des

accélérateurs, car c'est un émulateur d'exécution GPU. C'est cet accélérateur qui est utilisé si vous choisissez de debugger le code noyau. Nous reviendrons plus en détail sur la manière de debugger du code C++ AMP dans un prochain article.

L'accélérateur CPU, `accelerator::cpu_accelerator`, est seulement utilisé dans un cadre avancé de transfert de données depuis le noyau vers l'host que l'on appelle aussi « **staging array** ». L'utilisation de cet accélérateur CPU dépasse le cadre de cette introduction. Vous pouvez afficher facilement toutes les propriétés des accélérateurs disponibles sur votre machine, il vous suffit de frapper ces quelques lignes de code :

```
std::for_each(accs.begin(), accs.end(), [&](accelerator acc)
{
    std::wcout << «New accelerator: » << acc.description << std::endl;
    std::wcout << «device_path = » << acc.device_path << std::endl;
    std::wcout << «version = » << (acc.version >> 16) << '.' << (acc.
version & 0xFFFF) << std::endl;
    std::wcout << «dedicated_memory = » << acc.dedicated_memory << « KB»
<< std::endl;
    std::wcout << «doubles = » << ((acc.supports_double_precision) ?
«true» : «false») << std::endl;
    std::wcout << «limited_doubles = » << ((acc.supports_limited_
double_precision) ? «true» : «false») << std::endl;
    std::wcout << «has_display = » << ((acc.has_display) ? «true» :
«false») << std::endl;
    std::wcout << «is_emulated = » << ((acc.is_emulated) ? «true» :
«false») << std::endl;
    std::wcout << «is_debug = » << ((acc.is_debug) ? «true» : «false»)
<< std::endl;
    std::wcout << std::endl;
});
```

Si j'exécute ce code sur ma machine, j'obtiens les résultats suivants : [Fig.10]. On retrouve tous les types d'accélérateurs et leurs propriétés respectives. Ces informations sont très utiles si vous souhaitez connaître les capacités de vos accélérateurs. En général, c'est l'accélérateur attaché à votre carte graphique qui vous importe, car c'est généralement celle-ci qui offrira le plus de performances. Si nous revenons à notre démonstration, nous n'avons jusqu'à présent jamais présenté un exemple qui utilisait un accélérateur. Et pourtant, il y a toujours un accélérateur utilisé. Dans le code ci-dessous, nous faisons apparaître l'usage d'un accélérateur.

```
accelerator pick_accelerator()
{
    accelerator rc;
    vector<accelerator> accelerators = accelerator::get_all();
    auto it = std::find_if(accelerators.begin(), accelerators.
end(), [&](accelerator accel)
    { return !accel.is_emulated; });

    if (it != accelerators.end()) rc = *it;

    return rc;
}

void do_it(int* v, int size, int to_add)
```

```
{
    accelerator acc = pick_accelerator();

    array_view<int> av(size, v);

    parallel_for_each(acc.default_view, av.extent, [=](index<1> idx)
restrict(amp)
    {
        av[idx] += to_add;
    });
}
```

Vous noterez que la méthode `parallel_for_each` ne prend pas en paramètre un accélérateur, mais une vue sur un accélérateur : `accelerator_view`. Le rôle de la classe `accelerator_view` est d'encapsuler l'accélérateur afin de le protéger des différents usages multithreads dans votre code. Pour conclure cette parenthèse sur les accélérateurs, ils apparaissent généralement lorsque vous souhaitez gérer plusieurs cartes graphiques pour paralléliser vos traitements sur plusieurs cartes graphiques.

> Array

Nous avons utilisé le type `array_view` qui se prête bien à notre exemple, mais il existe un autre type de conteneur, qui cette fois n'est pas wrapper, mais un « vrai » conteneur, `array<T, N>`. D'un point de vue de l'usage, le type `array<T, N>` est un peu différent d'un `array_view`, mais l'instanciation est équivalente. La capture vis-à-vis de l'expression lambda de la méthode `parallel_for_each` change : contrairement à l'instance `array_view`, le type `array` n'est pas un wrapper sur des données, mais un vrai conteneur qui ne doit pas être passé par valeur, mais par référence. En revanche, la variable `to_add` doit être passée par valeur, ce qui nous donne ce type de capture au niveau de l'expression lambda `[=,&arr]`.

```
void do_it(int* v, int size, int to_add)
{
    accelerator acc = pick_accelerator();

    array<int> arr(size, v);

    parallel_for_each(acc.default_view, arr.extent, [=,&arr](index
```

```
C:\Windows\system32\cmd.exe
New accelerator: NVIDIA Quadro 2000M (Microsoft Corporation - UDDM v1.2)
device_path = PCI\VEN_10DE&DEV_0DDA&SUBSYS_21D1170A&REV_A1\4&1DE8C1ED&0&0000
version = 11.0
dedicated_memory = 2047552 KB
doubles = true
limited_doubles = true
has_display = false
is_emulated = false
is_debug = false

New accelerator: Microsoft Basic Render Driver
device_path = direct3d\warp
version = 11.1
dedicated_memory = 0 KB
doubles = false
limited_doubles = false
has_display = false
is_emulated = true
is_debug = false

New accelerator: Software Adapter
device_path = direct3d\ref
version = 11.1
dedicated_memory = 0 KB
doubles = true
limited_doubles = true
has_display = true
is_emulated = true
is_debug = false

New accelerator: CPU accelerator
device_path = cpu
version = 0.1
dedicated_memory = 3964140 KB
doubles = false
limited_doubles = false
has_display = false
is_emulated = true
is_debug = false

Press any key to continue . . .
```

Fig.10

```
<1> idx) restrict(amp)
{
    arr[idx] += to_add;
});

copy(arr, v);
}
```

Une fois la boucle **parallel_for_each** lancée, nous devons attendre le retour du traitement sur la carte graphique puis copier les données dans le buffer d'origine nous-mêmes, une fois la boucle terminée. Vous l'aurez compris, la fonction `copy` assure exactement ce comportement.

> L'exemple fil rouge : la multiplication matricielle

Toutes les introductions sur la programmation GPU utilisent l'algorithme de multiplication matricielle, car il est à la fois simple et démontre efficacement les performances du calcul sur un GPU.

Voici un rappel de l'algorithme via une feuille Excel : [Fig.11]. L'objectif est de calculer le produit de deux matrices **A** et **B** et d'engendrer la matrice produit **C**. Dans notre exemple la matrice **A** est de dimension **M** x **W**, avec **M**=2 et **W**=4, la matrice **B** est dimension **W** x **N** avec **W**=4 et **N**=6 et enfin la matrice produit **C** est de dimension **M** x **N**. Chaque ligne **A** est multipliée par toutes les colonnes entières de **B**. Chaque multiplication est sommée dans une variable temporaire. Après avoir multiplié tous les éléments de la ligne avec tous les éléments de la colonne courante. Le résultat temporaire est placé dans la cellule correspondante de la matrice produit **C**.

> Implémentation

Dans Visual Studio 2012, vous créez un projet C++ en choisissant le modèle « **Empty Project** », puis vous ajoutez un fichier C++. Le code ci-dessous représente une implémentation de la multiplication matricielle.

```
#include <iostream>
#include <vector>

using std::vector;

void MatrixMultiply(vector<int>& vC, vector<int>& vA, vector<int>& vB, int M, int N, int W)
{
    for (int row = 0; row < M; row++)
        for (int col = 0; col < N; col++)
        {
            int sum = 0;
            for (int k = 0; k < W; k++)
                sum += vA[row * W + k] * vB[k * N + col];
            vC[row * N + col] = sum;
        }
}

void main()
{
    // Dimensions des matrices
    int M = 2;
    int W = 4;
```

```
int N = 6;

// 3 vecteurs pour représenter les matrices
vector<int> A(M*W);
vector<int> B(W*N);
vector<int> C(M*N);

// Peupler la matrice A
for (int i = 0; i < A.size(); i++) A[i] = i + 1;

// Peupler la matrice B
for (int i = 0; i < B.size(); i++) B[i] = i + 1;

MatrixMultiply(C, A, B, M, N, W);
}
```

L'implémentation de l'algorithme est sans surprise et correspond à la description de l'algorithme. Nous allons, comme nous l'avons fait dans l'exemple « Hello World », introduire progressivement la parallélisation avec C++ AMP. Nous recopions l'algorithme original et renommons le nom de la méthode par **MatrixMultiplyAMP**. Nous devons ajouter le fichier « **amp.h** » ainsi que l'espace de nom « **concurrency** », puis nous introduisons trois **array_view**.

```
#include <vector>
#include <amp.h>
using std::vector;
using namespace concurrency;

void MatrixMultiplyAMP(vector<int>& vC, vector<int>& vA, vector<int>& vB, int M, int N, int W)
{
    array_view<int> a(M*W, vA), b(W*N, vB);
    array_view<int> c(M*N, vC);
    for (int row = 0; row < M; row++)
        for (int col = 0; col < N; col++)
        {
            int sum = 0;
            for (int k = 0; k < W; k++)
                sum += a[row * W + k] * b[k * N + col];
            c[row * N + col] = sum;
        }
}
```

MatrixSimple - Excel

FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW Load Test Team Bruno Bo...

L10 : =((C10*L3)+(D10*L4)+(E10*L5)+(F10*L6))

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
2	A * B = C				B									
3									1	2	3	4	5	6
4		A => 2 x 4 (M = 2, W = 4)							7	8	9	10	11	12
5		B => 4 x 6 (W = 4, N = 6)							13	14	15	16	17	18
6		C => 2 x 6 (M = 2, N = 6)							19	20	21	22	23	24
9	A				C									
10		1	2	3	4			col 0	col 1	col 2	col 3	col 4	col 5	
11		5	6	7	8			row 0	130	140	150	160	170	180
12								row 1	290	316	342	368	394	420
13								C(0,3)	160 = (1 * 4 + 2 * 10 + 3 * 16 + 4 * 22)					

Fig.11

À ce stade, le code compile correctement, mais cependant, il n'est pas encore parallélisé. Nous passons le code à une dimension à deux dimensions. Puis nous remplaçons les deux boucles des lignes et des colonnes par la méthode **parallel_for_each**. Avec la variable **idx** de type **index**, nous réaffectons les variables **row** et **col**.

```
#include <vector>
#include <amp.h>
using std::vector;
using namespace concurrency;

void MatrixMultiplyAMP(vector<int>& vC, vector<int>& vA, vector<int>& vB, int M, int N, int W)
{
    array_view<int, 2> a(M,W, vA), b(W,N, vB);
    array_view<int, 2> c(M,N, vC);
    //for (int row = 0; row < M; row++)
    //for (int col = 0; col < N; col++)
    parallel_for_each(c.extent, [=](index<2> idx) restrict(amp)
    {
        int row = idx[0];
        int col = idx[1];
        int sum = 0;
        for (int k = 0; k < W; k++)
            sum += a(row, k) * b(k, col);
        c(row, col) = sum;
    });
}
```

Cependant, il nous reste quelques modifications afin d'obtenir un code correct. Nous remplaçons dans le type du template pour les variables **a** et **b**, qui représentent les deux matrices d'entrées venant nourrir le traitement, du type **int** à **const int**, ce qui apporte une optimisation au traitement. La variable **c**, représentant la matrice produit qui va recevoir les résultats, on appelle la méthode **discard_data()** afin de préciser que les valeurs en entrées de cet **array_view**, ne sont pas à copier sur le GPU, ceci ajoute une touche d'optimisation.

```
void MatrixMultiplyAMP(vector<int>& vC, vector<int>& vA, vector<int>& vB, int M, int N, int W)
{
    array_view<const int,2> a(M, W, vA);
    array_view<const int,2> b(W, N, vB);
    array_view<int,2> c(M, N, vC);
    c.discard_data();

    parallel_for_each(
        c.extent, [=](concurrency::index<2> idx)
        restrict(amp)
    {
        int row = idx[0];
        int col = idx[1];
        int sum = 0.0f;

        for(int k = 0; k < W; ++k)
            sum += a(row, k) * b(k, col);
    });
}
```

```
c[idx] = sum;
});

c.synchronize();
}
```

Nous avons à ce stade à disposition deux versions de l'algorithme, une en mode séquentiel et la seconde en mode parallèle. Pour mesurer les performances, il nous faut un minimum d'outillage pour mesurer les performances de traitements reposant sur la technologie GPU. Vous trouverez sur le blog Microsoft « **Parallel Programming in Native Code** », un billet sur « **High-resolution timer for C++** » accompagné des sources de la classe **Timer**, <http://blogs.msdn.com/b/nativeconcurrency/archive/2011/12/28/high-resolution-timer-for-c.aspx>, qui correspond parfaitement à notre besoin.

Pour observer les performances de C++ AMP, nous avons révisé le code et introduit l'utilisation de la classe **Timer** via « **timer.h** » : voir les sources de l'article sur notre site.

Nous obtenons ainsi des matrices carrées bien plus lourdes de taille 1024*1024, afin de montrer les gains obtenus avec la technologie GPU. Naturellement, en fonction de votre matériel vous obtiendrez des résultats différents. Pour le calcul en mode séquentiel, le traitement a pris environ 9 secondes, alors que le calcul parallèle avec C++ AMP a pris 142 millisecondes.

En résumé

Pour conclure, je vous propose de dresser le bilan des éléments que vous avez appris.

Sur le plan conceptuel :

- Sur des volumes de données immenses, le calcul sur GPU offre des performances extraordinaires
- Microsoft souhaite démocratiser la programmation GPU avec C++ AMP sur toutes les plateformes Windows et non Windows

Sur le plan de la programmation :

- class **index<N>** : pointe une valeur dans un conteneur multidimensionnel
- class **extent<N>** : taille d'un conteneur multidimensionnel
- **restrict(amp, cpu)** : restreint la syntaxe C++ pour exécuter une méthode ou une expression lambda en mode noyau
- **parallel_for_each** : exécute une boucle en parallèle sur des données multidimensionnelles
- class **array_view<T,N>** : wrapper multidimensionnel sur des données
- class **accelerator** : accélérateur utilisé en interne pour exécuter le code parallèle
- class **accelerator_view** : isole l'accélérateur des différents usages multithreads
- class **array<T,N>** : conteneur multidimensionnel sur des données.

Avec ces quelques éléments, vous pouvez déjà produire des programmes utilisant la librairie C++ AMP et obtenir des exécutions très performantes. En adressant la programmation GPU avec autant d'abstractions, vous comprenez qu'il n'est pas nécessaire d'être un super expert pour paralléliser du code sur GPU. J'espère que cette découverte de C++ AMP, vous a mis l'eau à la bouche, et que vous allez vous essayer à construire des applications utilisant C++ AMP.



Bruno Boucard
MVP - Architecte Finaxys



Koneki, le développement M2M simplifié

Le projet Koneki vient tout juste de souffler sa première bougie, et fait déjà partie de la livraison simultanée annuelle Juno. Nous vous proposons à travers ces quelques pages de découvrir les bases du « Machine-to-Machine » (M2M) et les fonctionnalités les plus intéressantes de Koneki.

Vous avez dit...Koneki ? « Koneki », en Maori, signifie « ce lieu ». En appelant le projet Koneki, nous avons souhaité signifier que les développeurs M2M pourront très bientôt bénéficier d'outils s'intégrant en toute transparence à l'environnement Eclipse pour développer, simuler et tester leurs applications en toute simplicité. Aujourd'hui, lorsqu'une entreprise se lance dans un projet de développement M2M, elle se retrouve très rapidement confrontée à une foule de problèmes d'ingénierie – développement des applications embarquées, des applications mobiles ou Web, manipulation des protocoles industriels – et perdue dans le labyrinthe des outils permettant de les résoudre. Le but de Koneki est de les guider vers la sortie, en regroupant tous les outils nécessaires en un même lieu.

> Introduction au « Machine-to-Machine »

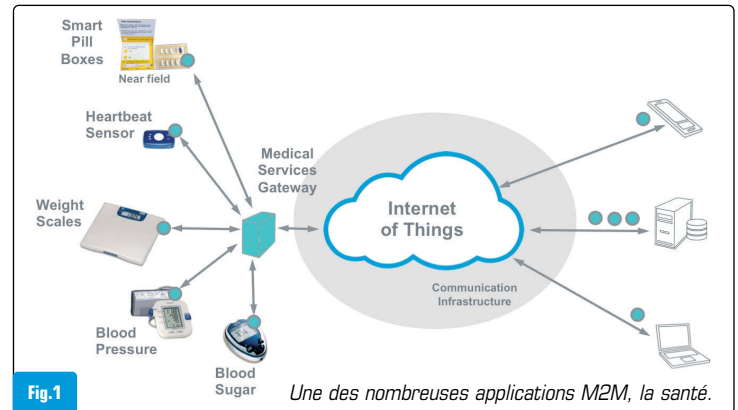
Le terme « *Internet of Things* » (IoT) ne vous est sans doute pas inconnu. Celui de « M2M » repose sur la même idée, celle de connecter tous les objets possibles et imaginables à l'Internet, et de leur offrir la possibilité à la fois de partager leurs données et d'être commandés à distance.

D'ici 2020, on estime à environ 50 milliards le nombre d'appareils matériels connectés dans le monde.

Les domaines du quotidien où le M2M modifie la manière dont nous interagissons avec les objets sont déjà nombreux, et se multiplient de jour en jour. Celui de la médecine, illustré figure 1, en est un bon exemple. Le fait de pouvoir relier des choses telles que des tensiomètres électroniques ou des électrocardiogrammes à l'Internet, permet d'un côté aux développeurs de créer des applications Web ou mobiles autour des données recueillies, et de l'autre aux fabricants de ces appareils de les gérer à distance, par exemple pour les mettre à jour [Fig.1]. Le problème est que les solutions M2M aujourd'hui déployées s'appuient sur des technologies et protocoles de communication propriétaires, qui constituent un obstacle de taille à la croissance du marché M2M...

> Les acteurs du M2M à la rescousse

A la fin de l'année dernière, plusieurs acteurs de l'écosystème M2M ont décidé de prendre les choses en main en créant un Groupe de travail industriel (*Industry Working Group*) sous l'égide de la fondation Eclipse, pour adresser le plus gros problème actuel des solutions



M2M : leur manque d'interopérabilité. L'objectif du M2MIWG (*M2M Industry Working Group*) est de travailler à sa résolution, en fournissant notamment toutes les bibliothèques de communication, cadres applicatifs (*application framework*) et outils nécessaires à un développement simple et sain de solutions M2M [Fig.2]. A l'heure actuelle, deux projets « Eclipse Technology » sont pilotés par le M2M Industry Working Group : *Paho* et *Koneki*. Le but du projet *Paho* est de simplifier l'implémentation de protocoles légers pouvant servir aux applications M2M et IoT. Les contributions initiales ont émané d'IBM et Eurotech sous la forme de bibliothèques client C et Java pour le protocole MQTT. MQTT (*Message Queue Telemetry Transport*) est un protocole de messagerie TCP/IP permettant d'optimiser l'utilisation de la bande passante lors des communications M2M. Pour reprendre ce qui a été dit en introduction, l'objectif de Koneki est de fournir des outils de développement simples pour toute la chaîne M2M, de la création d'applications embarquées à la simulation de scénarios de communication, en passant par le développement d'applications mobiles ou Web.

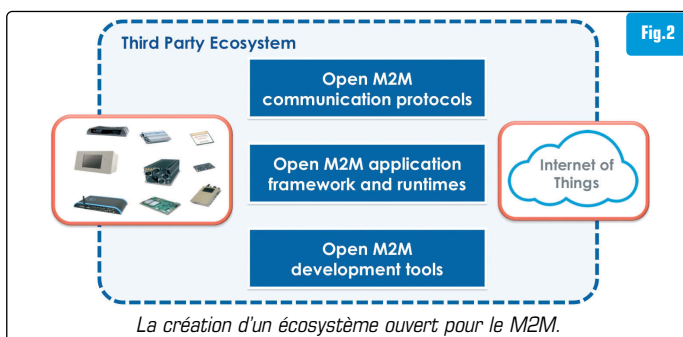
> Lua, un langage de programmation aussi puissant que léger

Le langage de programmation Lua (prononcez « lou-ah ») est né au Brésil en 1993 du besoin de disposer d'un langage facile à intégrer dans d'autres applications. Ses concepteurs l'ont donc pensé « léger » et en pur ANSI-C, ce qui le rend on ne peut plus facile à transporter vers d'autres plateformes aussi petites que des microcontrôleurs limités à quelques centaines de kilo-octets de RAM et de mémoire Flash. La distribution eLua [1], va encore plus loin dans l'optimisation pour les cibles à plus fortes contraintes.

Le langage Lua s'apprend très facilement. De nombreux éditeurs de jeux vidéo et de plateformes de développement mobile en ont même fait le langage de développement privilégié pour leurs applications.

> Lua pour le développement d'applications M2M

Vous vous demandez sans doute pourquoi nous avons choisi comme composante principale de Koneki un IDE dédié à Lua, appelé *Lua Development Tools* (LDT), et non quelque chose de plus spécifique au



M2M. La réponse est simple : l'un des plus gros freins à la création de solutions M2M est, selon nous, la complexité de développement des applications embarquées dans les boîtiers de communication déployés sur le terrain.

Il faut bien garder à l'esprit que les domaines d'application du M2M sont infinis et que personne ne veut avoir à apprendre un langage aussi complexe que le C, tout cela pour, au final, ne manipuler que quelques structures de données à envoyer vers un serveur distant. Lua a déjà été adopté par les développeurs de jeux vidéo et applications mobiles, pourquoi ne pas l'adopter également pour le développement d'applications embarquées ?

Lua Development Tools a déjà été téléchargé et installé par plus de 15 000 utilisateurs !

> Premiers pas avec Lua Development Tools

Lua Development Tools (LDT), peut être installé depuis le dépôt officiel Juno (dans la catégorie « PROGRAMMING LANGUAGES ») ou depuis le *Eclipse Marketplace*. Le plus simple pour démarrer est de créer un nouveau projet en lançant l'assistant de création par **FILE > NEW > OTHER... > LUA PROJECT**. Vous avez le choix entre démarrer un nouveau projet, ou vous baser sur des sources Lua existantes. Une fois la création terminée, il vous sera proposé de basculer vers la perspective Lua (qu'il est conseillé d'utiliser), et le tour est joué !

> Un effort particulier porté à l'assistance utilisateur

Afin que les développeurs pas forcément habitués à programmer d'emblée ne soient pas trop vite rebutés, Koneki propose un support avancé du langage Lua. Lua est un langage dynamique, ce qui rend délicate la mise en place de fonctionnalités telles que l'auto complétion ou la navigation dans le code, quand il n'y a ni typage, ni distinction claire entre variables simples et fonctions, etc. LDT s'appuie sur DLTk (*Dynamic Languages Toolkit*), qui fournit des fonctionnalités de base pour le développement du code, mais il nous a fallu améliorer son analyse syntaxique pour permettre à l'IDE d'inférer un maximum d'informations de typage.

Cette fonctionnalité a pu voir le jour, notamment grâce au travail effectué sur la définition d'un langage de documentation, qui permet de décrire de façon détaillée les API des modules Lua. Les utilisateurs bénéficient ainsi non seulement d'une documentation comparable à Javadoc, avec vues et pop-ups, mais aussi de l'auto complétion et de la possibilité de parcourir le code (pour se rendre rapidement à la déclaration d'une méthode par exemple). Le Listing 1 vous donne un aperçu de la documentation du module standard 'io' de Lua 5.1, et la Figure 3 illustre la fonctionnalité d'auto complétion.

Listing 1

```

-- Input and Output Facilities.
-- The I/O library provides function for file manipulation.
-- @module io

-- a file handle.
-- @type #file

-- This function opens a file, in the mode specified in the

```

```

string `mode`. It
-- returns a new file handle, or, in case of errors, nil plus
an error message.
-- The `mode` string can be any of the following:
-- «r»: read mode (the default);
-- «w»: write mode;
-- «a»: append mode;
-- «r+»: update mode, all previous data is preserved;
-- «w+»: update mode, all previous data is erased;
-- «a+»: append update mode, previous data is preserved, writing is only
-- allowed at the end of file.
-- The `mode` string can also have a 'b' at the end, which is needed in
-- some systems to open the file in binary mode. This string is exactly what
-- is used in the standard C function `fopen`.
-- @function [parent=#io] open
-- @param filename
-- @param mode
-- @return #file

-- Equivalent to `file:close()`. Without a `file`, closes the default
-- output file.
-- @function [parent=#io] close
-- @param file

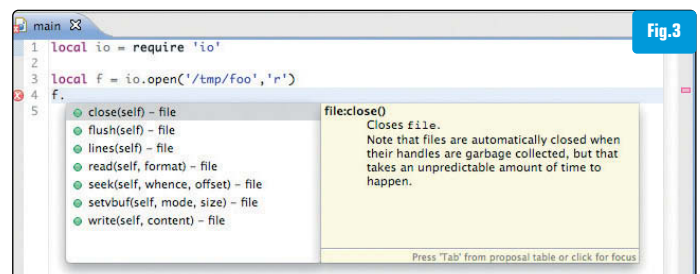
-- [...]

```

Il est également possible d'utiliser un mécanisme similaire aux *target platforms* de PDE (*Plugin Development Environment*), qui permet de stocker dans un fichier ZIP la définition des API Lua visibles pour un développeur. Koneki ne livre pour l'instant aucun de ces fichiers appelés « Environnements d'exécution » ; en revanche, l'environnement d'exécution Lua 5.1 ainsi que celui de l'environnement de développement mobile Gideros sont d'ores et déjà disponibles sur le GitHub de Sierra Wireless [2].

> Déboguer une application Lua

Outre son mécanisme d'auto complétion, Lua Development Tools 0.8 s'accompagne d'un débogueur utilisable avec la presque totalité des machines virtuelles Lua. Ce débogueur prend la forme d'un client DBGp (DeBuG protocol) écrit en Lua qui instrumente un script Lua pour le faire communiquer avec un serveur de débogage exécuté dans l'IDE. Pour utiliser le débogueur, il vous suffit d'inclure deux fichiers (*debugger.lua* et *debugintrospection.lua*) dans votre LUA_PATH, puis de lancer le client de debug en précédant le code à déboguer d'un appel à : `require('debugger')()`. Vous pouvez également lancer le débogueur depuis n'importe quel script existant via la ligne de commande suivante : `lua -e «require('debugger')();»`



Auto complétion lors de l'utilisation du module 'io' de Lua 5.1.

MyApp.lua. Le débogage ne fonctionne que si un serveur est lancé au préalable dans l'IDE, une *debug configuration* doit donc être lancée avant d'exécuter le script Lua instrumenté. La création d'une configuration de débogage s'effectue par le menu habituel **Run > DEBUG CONFIGURATIONS...**

Si votre machine virtuelle Lua ne sort pas trop de l'ordinaire – elle doit au moins implémenter la méthode *loadstring()* – toutes les fonctionnalités auxquelles vous êtes habitué en tant que développeur C ou Java dans Eclipse vous seront accessibles : points d'arrêt (y compris les points d'arrêt conditionnels), recherche et modification de variables, expressions interactives, débogage *multi-thread* (coroutines)... [Fig.4].

> Les autres outils Koneki

Simulateur OMA-DM — Nous avons mis à disposition début 2012 un plug-in Eclipse permettant de configurer et d'exécuter des simulations OMA-DM. OMA-DM est un standard de communication extrêmement courant dans les télécommunications, qu'on rencontre notamment dans la surveillance et la synchronisation d'appareils tels que les téléphones mobiles (ou tout autre module de communication intervenant dans la chaîne M2M). L'interface utilisateur du simulateur comprend un formulaire permettant d'éditer les paramètres de la simulation, ainsi qu'une représentation graphique de l'arborescence de l'appareil simulé. Il comprend également un tableau de bord interactif permettant de visualiser les paquets OMA-DM échangés entre client et serveur. Le simulateur OMA-DM disponible dans Koneki est compatible avec l'implémentation open source de serveur OMA-DM disponible dans le projet *Funambol Device Management* [3]. Si vous désirez l'essayer, notre Manuel utilisateur contient toutes les informations nécessaires à l'installation de ce dernier.

Modèle M2M — Nous travaillons actuellement à la définition d'un métamodèle d'applications M2M dont le but est d'explicitier de manière claire certaines notions importantes : données manipulées par les applications, protocoles utilisés lors des échanges de données entre différents éléments de la chaîne M2M, dépendances vis-à-vis de matériel spécifique (GPS, par exemple) ou de bibliothèques externes, etc. Nous espérons qu'un tel modèle encouragera la création d'outils extensibles, en permettant à chacun de développer ses propres extensions, par exemple un outil utilisant les informations

renseignées dans le modèle d'application pour estimer la bande passante consommée avec tel ou tel protocole de communication. Un tel modèle a aussi vocation à être utilisé au moment de l'exécution de l'application, en configurant par exemple automatiquement une API d'accès aux données cachant la complexité de protocoles de communication sous-jacents. La version initiale de ce modèle M2M, axée principalement sur la description des données manipulées par une application, est disponible dans le dépôt Git [4] du projet. Elle adresse les spécificités du M2M, en opposant par exemple clairement les données uniquement *envoyées* vers d'autres machines aux données pouvant aussi être *mises à jour* à distance, ou encore en décrivant les opérations exécutables à distance. Les exigences de ce méta modèle M2M sont discutées au sein du M2M Industry Working Group, qui s'appuie sur des cas industriels concrets pour identifier les notions fondamentales devant être impérativement couvertes.

> À suivre...

Vous devez sans doute vous demander maintenant ce que réservent les nouvelles versions de Koneki, et surtout quel nouveau contenu M2M nous allons ajouter à Koneki.

Notre prochaine étape sera de contribuer au développement d'une plateforme embarquée Lua, compatible Linux, destinée à simplifier la gestion à distance du système, la manipulation des entrées/sorties, les accès au réseau, etc. Au-delà du M2M industriel, nous cherchons aussi à répondre à la demande de la communauté *DIY* (Do It Yourself), en proposant un accès homogène et plus direct aux API de communication (SMS, connectivité 3G, etc.), en proposant des fonctionnalités de gestion à distance des systèmes embarqués (mise à jour de firmware *over-the-air*, suivi de l'état du réseau, etc.), ... C'est l'objectif du projet Mihini [5], récemment proposé à la fondation Eclipse. Nous envisageons également de fournir un environnement de modélisation complet dédié au M2M, développé en partenariat avec l'équipe Damos [6]. Damos est un tout nouveau projet Eclipse, qui porte sur la création d'un environnement de développement permettant de concevoir des systèmes embarqués en suivant une démarche orientée *dataflow*. Nous travaillons actuellement sur le plan de la version 0.9 de Koneki, qui devrait être déjà disponible. Vos suggestions sont plus que les bienvenues, alors venez nombreux déposer vos commentaires sur notre forum [7] ou notre liste de diffusion [8] ! Nous projetons une livraison de la version 1.0 en 2013, en même temps que Kepler... les prochains mois s'annoncent chargés ! — www.eclipse.org/koneki

Références

- [1] <http://www.eluaproject.net>
- [2] <http://sierrawireless.github.com/lua-execution-environments>
- [3] <https://dm.forge.funambol.org>
- [4] <http://git.eclipse.org/c/koneki>
- [5] <http://www.eclipse.org/proposals/technology.mihini>
- [6] <http://www.eclipse.org/proposals/tools.damos>
- [7] <http://www.eclipse.org/forums/eclipse.koneki>
- [8] <http://dev.eclipse.org/mhonarc/lists/koneki-dev>



Benjamin Cabé
évangéliste Open Source chez Sierra Wireless.
Il est passionné par Eclipse et son écosystème, et anime le projet et la communauté Koneki.
Blog : <http://blog.benjamin-cabe.com> et @kartben sur Twitter.

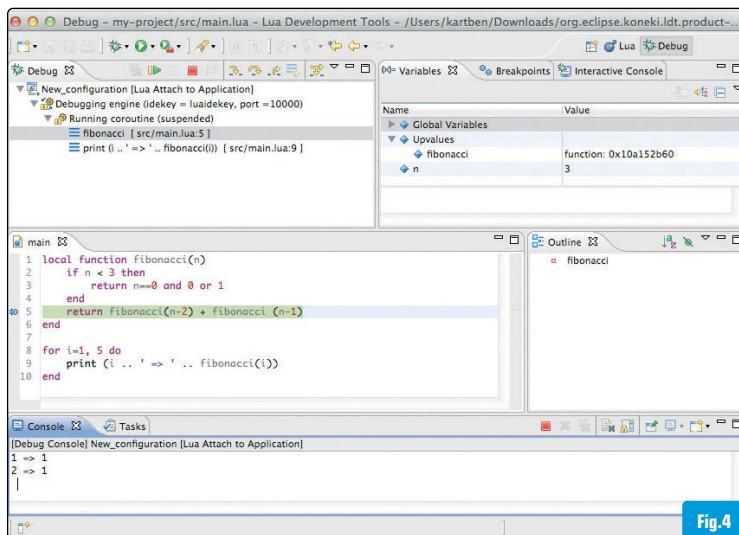


Fig.4

Le débogueur Lua en action.

JavaScript : la révolution est-elle en marche ?

2^e partie



Nous avons abordé dans la première partie de notre dossier, plusieurs éléments autour de JavaScript, TypeScript et Node.JS. Nous continuons ce mois-ci l'exploration du monde merveilleux de JavaScript avec plusieurs avis de développeurs et des articles techniques ! Enjoy !

François Tonic

Créer un service Cloud en moins de 500 lignes

En voici un titre racoleur ! Du moins dans le monde de l'IT... J'espère ne pas lancer une mode qui s'apparenterait aux solutions pour maigrir en moins de 7 jours ou d'apprendre le Javanais en moins d'un mois, mais je relève néanmoins le défi énoncé dans le titre. Par la formulation "créer un service", je souhaite proposer l'optique du service dans son ensemble. Cela comprend le langage de programmation, les frameworks employés mais aussi le système de stockage des données, la diffusion du service en termes de plateformes et enfin son hébergement. Pour être bref, l'IT évolue beaucoup actuellement. Nous étions encore récemment concentrés sur les nouveaux frameworks pour l'infrastructure en place. Aujourd'hui, toutes les briques sont remises en cause : langages de programmation (Scala, Groovy, JavaScript, Dart, TypeScript...), bases de données (MongoDB, Neo4j, Redis...), et l'hébergement avec tout ce qui touche au Cloud. En somme, sans oser avancer que cette application est celle du futur, aborder ce que pourrait être l'application du futur nécessite une perspective très large.

> Qu'est-ce qu'un service Cloud

Comme beaucoup de notions marketing de l'IT, le Cloud est certainement plus une idée qu'une technologie à proprement parler. Plutôt

que de le définir de façon absolue, je proposerai ici ma vision, ou du moins celle qui a guidé le design de l'application. Un service Cloud est un service accessible via Internet sur un nombre important de plateformes : ordinateurs, tablettes et smartphones (je n'ai volontairement cité aucune marque ou système d'exploitation). Chaque utilisateur du service peut y trouver un espace qui lui est personnel et communiquer avec les autres utilisateurs du service. Lorsqu'il passe d'un appareil à un autre pour utiliser le service, il y retrouve son espace. La traduction technique de ce cahier des charges correspond à un service proposant une interface accessible sur un maximum de plateformes (Web, Android, iOS...) proposant un hébergement et un stockage de données aux performances et au coût très extensibles afin d'être capable de gérer le succès du service (on ne sait jamais !).

> Le vif du sujet

Derrière un projet comme celui-là, il faut une idée de service. Pour cet article, je vous en propose une qui, a priori, n'est ni révolutionnaire, ni inintéressante, donc plutôt adaptée à son utilisation ici. Il s'agit de créer des espaces graphiques partagés, permettant d'échanger avec d'autres utilisateurs avec des dessins. Je l'ai sim-

plement appelé "Shared Whiteboard". Le service répond aux exigences du Cloud par la persistance du contenu des "tableaux", la diffusion du service sur le Web, les tablettes et les mobiles (les interfaces tactiles permettent de dessiner avec le doigt). Les échanges se font par le partage d'un même "tableau".

> Parlons technique

Nous voici arrivés au moment de choisir tous les composants techniques de notre service. Bien sûr, chaque choix pourrait être soumis à débat, je vous décrirai simplement les choix qui ont été faits et leurs justifications.

> Côté serveur

Le serveur n'a pratiquement aucune règle de gestion à implémenter, son rôle sera uniquement de recevoir les modifications sur les tableaux et de dispatcher ces modifications aux autres clients connectés au même tableau et au système de stockage. Node.js apparaît alors comme un choix tout désigné : très performant sur de petites tâches d'I/O, simple à héberger et très facile d'utilisation pour faire des WebSockets. Quelques lignes Node.js pour démarrer un serveur HTTP (avec express.js)

```
var _ = require('underscore');
var express = require('express');
...
var expressServer = express.createServer()
expressServer.use(express.bodyParser());
expressServer.use(express.errorHandler());
expressServer.use(express.static(__dirname + '/public'));
...
var port = parseInt(process.env.PORT, 10) || 1337;
expressServer.listen(port);
console.info('listen port ', port);
```

Pour communiquer avec l'interface Web et sachant qu'il faudra du Push, c'est bien évidemment Socket.io que nous allons utiliser. Configurer, connecter et transmettre les événements de modification des tableaux se font, encore une fois, quelques lignes.

```
var io = socketio.listen(expressServer, {
  «log level»: 2
});
...
io.sockets.on('connection', function (socket) {
  socket.on('connect', function(data) {
    ... // void ci-dessous
  });
  socket.on('clear', function(data) {
    ... // void ci-dessous
  });
  socket.on('element', function(data) {
    console.log('element message, receiving new', data.type,
      «element for», data.whiteboard, «whiteboard»);
    socket.broadcast.emit('element', data);
    ... // void ci-dessous
  });
});
```

Côté stockage, nous pouvons facilement identifier nos tableaux pour être des documents MongoDB avec une intégration Node.js rapide

et un hébergement facile également. Le fonctionnement asynchrone de MongoDB permet de s'intégrer rapidement dans les événements socket.io déjà préparés :

```
var mongo = mongoose.createConnection('mongodb://whiteboard:
***@alex.mongohq.com:10007/shared-whiteboard');
...
var WhiteBoardElement = mongo.model('WhiteBoardElement', new mon
goose.Schema({
  whiteboard: String,
  type: String
}));

io.sockets.on('connection', function (socket) {
  socket.on('connect', function(data) {
    WhiteBoardElement.find({
      whiteboard: data.whiteboard
    }, function(err, docs) {
      if(!err) {
        socket.emit('init', docs);
        console.log('connect message, whiteboard', data.whiteboard,
          «return», docs.length, «docs»);
      } else {
        console.error('connect message, error from mongodb', err);
      }
    });
  });
  socket.on('clear', function(data) {
    console.log('clear message, deleting elements for whiteboard',
      data);
    WhiteBoardElement.find({whiteboard: data}).remove();
  });
  socket.on('element', function(data) {
    console.log('element message, receiving new', data.type, «element
for», data.whiteboard, «whiteboard»);
    socket.broadcast.emit('element', data);
    var instance = new WhiteBoardElement(data, false);
    instance.save(function(err) {
      if(err) {
        console.error('error on adding the new element in mongodb',
          err);
      }
    });
  });
});
```

Nous voilà avec un code serveur complet pour seulement 54 lignes, et facilement hébergeable : <https://github.com/Swiip/shared-whiteboard/blob/master/app.js>

> Côté client

Avec 500 lignes et l'objectif de cibler un maximum de plateformes mobiles, autant dire que les applications codées en natif pour chaque plateforme sont totalement inaccessibles. L'interface générique avec des technologies Web est donc incontournable. Il va falloir également un framework de mise en forme qui s'adapte à tous les devices en un seul code. jQuery Mobile est pensé pour cet objectif, c'est lui que nous allons utiliser. Nous avons besoin de trois pages :

Accueil, Tableau et Options que nous mettons dans un seul document HTML comme le propose l'outil. Voir le code complet sur notre site. Le tableau en lui-même doit être une zone graphique. Dans les technologies Web, cela se concrétise par l'utilisation des nouveaux composants HTML5 SVG ou Canvas. Dans ce cas je me suis dirigé vers le framework SVG RaphaelJS et une petite librairie construite sur RaphaelJS qui s'appelle Raphael Sketchpad et qui permet justement de tracer au doigt ou à la souris des traits avec Raphael.

<http://ianli.com/sketchpad/http://ianli.com/sketchpad/>

Les puristes pourront mettre ma démonstration en défaut à cet endroit précis, et j'en ai conscience, je préfère de ce fait annoncer la chose clairement. Une librairie si peu répandue et dans laquelle en plus j'ai fait des modifications pourrait compter dans le nombre de lignes de code global, ce qui me ferait dépasser les 500 lignes (d'une centaine ou deux). Mais c'est néanmoins du code que j'ai réutilisé dans sa grande majorité, que je n'étais pas obligé de modifier en dehors de questions de performances et des photos (voir ci-dessous), et je suis loin d'avoir modifié autant de lignes que j'aurai de marge à la fin du compte, donc je me permettrai de continuer en considérant Sketchpad comme une librairie externe.

```
<head>
...
<script src="js/raphael.whiteboard.js"></script>
...
<script type="text/javascript">
...
function startWhiteboard() {
  window.whiteboard = Raphael.whiteboard("editor", {
    width: «100%»,
    height: «100%»,
    editing: true
  });
  whiteboard.addChangeHandler(function(attr) {
    if(listening) {
      attr.whiteboard = whiteboardName;
      socket.emit("element", attr);
    }
  });
}
...
</script>
</head>
```

Se connecter avec le serveur que nous avons créé précédemment demande d'utiliser l'interface cliente de socket.io. Ensuite, il s'agit d'écouter les modifications arrivant par la WebSocket et transmettre celles qui sont faites sur l'interface.

```
<head>
...
<script src="js/socket.io.min.js"></script>
...
<script type="text/javascript">
  var socket, whiteboardName, listening = true;

$(document).bind('pageinit', function(event) {
  socket = io.connect(«»);
```

```
...
});
...
function startWhiteboard() {
  ...
  socket.emit("connect", {
    whiteboard: whiteboardName
  });
  socket.on("init", function(elements) {
    listening = false;
    var result = whiteboard.paper().add(elements);
    listening = true;
  });
  socket.on("element", function(attr) {
    if(attr.whiteboard == whiteboardName) {
      listening = false;
      console.log("receiving new element of type «", attr.type);
      whiteboard.paper().add([attr]);
      listening = true;
    }
  });
}
</script>
</head>
```

Dans la définition du projet, j'avais parlé de distribution sur toutes les plateformes mobiles. En pratique, l'application actuelle fonctionne déjà sur la plupart des navigateurs mobiles mais il est possible d'aller plus loin en créant des applications susceptibles d'être distribuées sur les markets avec la solution PhoneGap. Pratiquement aucun code supplémentaire n'est nécessaire, il suffit de saisir l'adresse du repository GitHub dans : [\[https://build.phonegap.com/https://build.phonegap.com/\]](https://build.phonegap.com/https://build.phonegap.com/). Pour réellement intégrer l'application pour le mobile, on peut ajouter la fonctionnalité prendre en photo pour l'intégrer sur le tableau. Voir le code complet sur notre site.

Et nous voici au bas d'un document HTML très complet, qui fait appel à de nombreuses librairies, de nombreuses technologies mais qui ne fait pourtant que 172 lignes !

<https://github.com/Swiip/shared-whiteboard/blob/master/public/index.html>

> Conclusion

Un peu plus de 200 lignes en plus des modifications dans la librairie Sketchpad, nous sommes encore loin des 500 lignes, et pourtant les critères d'un service Cloud sont respectés.

Bien sûr, c'est un embryon de service, les bases techniques fonctionnent, mais nombre de questions se posent à l'intrépide qui voudra lancer son service Cloud : scalabilité (même si sur le papier tout est prévu), marketing, design, communication, rentabilisation.

Le but était de démontrer que les technologies actuelles permettent d'aller extrêmement loin en quelques lignes de code. Bien sûr, encore faut-il comprendre chaque technologie, s'en servir à bon escient et réussir à les intégrer. Les sources du projet sont toutes disponibles sur GitHub <https://github.com/Swiip/shared-whiteboard>

Le projet manque de documentation et de finalisation mais il n'est pas voué à devenir une référence en la matière. Toutefois, si certains s'y intéressent, je répondrais à toutes les questions s'y rapportant.

Matthieu Lux - Consultant JEE / Web Zenika

 zenika
ARCHITECTURE INFORMATIQUE

Implémentations d'applications Web avec Node et Express

Node permet notamment de mettre en œuvre des applications serveurs en utilisant le langage JavaScript. Une de ses spécificités (et de ses forces) réside dans son utilisation de JavaScript et du moteur V8 de Google afin de mettre à disposition une programmation asynchrone et non bloquante.

Node fournit également toute une interface de programmation pour différents cas d'utilisation dont HTTP afin d'implémenter des applications Web. Nous allons nous intéresser à ce type d'application tout au long de cet article. Comme pour les autres langages de programmation tels que Java et PHP, utiliser ces API « bas niveau » s'avère fastidieux et peu productif pour des applications réelles. Il apparaît donc judicieux de considérer des outils plus haut niveau afin de gagner en structuration, productivité et maintenabilité. Dans cet article, nous allons décrire comment utiliser Express (<http://http://expressjs.com/>), le plus populaire et utilisé des frameworks Web pour Node, afin de créer une petite application Web d'affichage de contacts. Ce framework fournit un intéressant cadre au-dessus des API HTTP de Node pour implémenter des applications Web suivant le modèle MVC (Model View Controller), une approche permettant de dissocier les traitements, des données manipulées et de leurs présentations dans l'interface graphique.

> Installation du framework Express

Nous n'allons pas revenir ici sur la manière d'installer Node. Il convient néanmoins de savoir que son installation met également à disposition un puissant outil de gestion de modules développés pour Node, NPM (Node Package Manager).

Nous allons l'utiliser afin d'installer les différents outils dont nous aurons besoin dans cet article. NPM se base sur un dépôt distribué contenant les modules ainsi que leurs descriptions et la liste des dépendances requises. Il permet ainsi de gérer de manière automatique l'installation de ces dernières. Les lecteurs ayant utilisé Linux ne seront pas dépayés car on y trouve des outils similaires tels que apt-get ou rpm.

Express est disponible dans le dépôt NPM et s'installe donc par l'intermédiaire de cet outil. Pour installer Express, il suffit simplement de taper la ligne de commande suivante :

```
$ npm install express
```

Comme nous utiliserons deux autres outils pour construire notre application Web, il faut également installer ceux-ci de manière similaire. Le premier est Jade, un moteur de templates pour Node, et Less, un préprocesseur facilitant l'utilisation de CSS. Les lignes de commande suivantes permettent d'installer ces deux outils :

```
$ npm install jade
$ npm install less
$ npm install less-middleware
```

Il est à noter que ces différents outils ont été installés dans le répertoire courant. NPM y crée un répertoire **node_modules** les contenant. Maintenant que tous les outils nécessaires ont été installés,

nous allons nous attaquer à notre application Web à proprement parler. Commençons par le squelette de celle-ci.

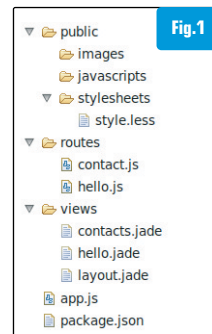
> Création du squelette de l'application

Nous pourrions bien sûr partir de zéro et mettre en place les différents éléments mais Express fournit un outil en ligne de commande pour nous aider dans cette tâche. Il est situé dans le répertoire **node_modules/express/bin** directement dans notre répertoire courant.

Cet outil nous permet de choisir les différents outils additionnels que nous allons utiliser dans notre application. La ligne de commande suivante sous-entend que nous utilisons Jade pour les templates ainsi que Less pour l'écriture des styles CSS :

```
$ ./node_modules/express/bin/express -c less contactapp
```

Il est à noter que le dernier paramètre précise le répertoire dans lequel sera créé le squelette de notre application. Nous l'appellerons



contactapp puisqu'elle retournera une liste de contacts.

L'organisation choisie est la suivante: un fichier **app.js** pour la configuration de l'application, un sous-répertoire **public** pour les ressources statiques, un sous-répertoire **routes** pour les contrôleurs et un sous-répertoire **views** pour les vues Jade. À noter la présence d'un fichier **package.json** à la racine contenant la description du module correspondant à l'application [Fig.1].

Décrivons maintenant plus en détail l'organisation de l'application ainsi que ses constituants.

> Organisation et configuration de l'application

Nous avons laissé Express créer pour nous le squelette de notre application. Nous allons dans cette section regarder comment celle-ci est organisée. Comme vous le savez, Node met en œuvre une organisation sous forme de modules. Nous allons faire quelques rappels à ce niveau.

Ces derniers peuvent se référencer les uns les autres par le biais de la fonction **require**, cette fonction permettant d'accéder aux objets mis à disposition par le module identifié par un nom. Trois types de noms sont utilisables :

- Les noms correspondant à des modules de Node lui-même tels que **http**, **filesystem**...
- Les noms correspondant à des modules présents dans le répertoire **node_modules** comme dans notre cas **express**, **jade** et **less**.

- Les noms référençant des fichiers JavaScript via un chemin explicite tels que «./routes/contacts». Dans ce cas, il est possible d'omettre l'extension du fichier JavaScript.

Il est possible, et recommandé, de tirer parti de cette fonctionnalité de Node afin d'organiser de manière optimale notre application Web. Une bonne pratique consiste également en l'utilisation d'un fichier JavaScript point d'entrée pour les applications. Ce point central pour notre application correspond au fichier **app.js** directement sous le répertoire de l'application. Ce fichier contient les traitements de base et sera exécuté via la commande Node pour lancer l'application. Il contient les différents traitements suivants :

- Importation des modules système, d'Express et de l'application;
- Initialisation d'Express;
- Configuration des routes pour accéder à l'application;
- Création du serveur hébergeant l'application.

Tout d'abord l'importation des modules nécessaires se réalise par l'intermédiaire de la fonction **require** de Node, comme décrit ci-dessous :

```
var express = require('express')
, contact = require('./routes/contact')
, http = require('http')
, path = require('path');
```

Il convient ensuite de créer une instance d'application Express avec la fonction **express** mise à disposition par le framework. Les configurations générales de cette application se font ensuite en utilisant les méthodes **set** et **use** de cette instance, la première pour le positionnement de propriétés et la seconde pour ajouter des middlewares à l'application prenant part à la chaîne de traitement des requêtes HTTP. Dans le code ci-dessous, nous définissons notamment le port d'accès de l'application ainsi que le répertoire contenant les fichiers des vues et le moteur de templates. Nous activons ensuite, entre autres, les supports de routage, des ressources statiques, la compression automatique des contenus.

```
var app = express();

app.configure(function(){
  app.set('port', process.env.PORT || 3000);
  app.set('views', __dirname + '/views');
  app.set('view engine', 'jade');
  app.use(express.favicon());
  app.use(express.logger('dev'));
  app.use(express.compress());
  app.use(express.bodyParser());
  app.use(express.methodOverride());
  app.use(app.router);
  app.use(require('less-middleware')({ src: __dirname + '/public' }));
  app.use(express.static(path.join(__dirname, 'public')));
});

app.configure('development', function(){
  app.use(express.errorHandler());
});
```

Comme vous pouvez le constater, les appels des méthodes **set** et **use** sont encapsulés dans des fonctions en paramètre de la méthode **configure**. Cette façon de faire offre des facilités afin d'activer ou non des configurations en fonction de l'environnement dans lequel est exécutée l'application (tous, développement, production). Dans notre cas, la détection et l'affichage des erreurs sont activés uniquement pour le développement.

Passons maintenant aux routes pour accéder aux pages. Leurs définitions dans Express se réalise très simplement puisqu'elles permettent d'associer un chemin relatif avec une fonction de traitement pour une méthode HTTP donnée, comme décrit ci-dessous. Le choix de la méthode HTTP est déterminé par la méthode utilisée lors de

l'association, dans notre cas **get**. Ainsi, l'exemple suivant spécifie que la fonction **contact.list** aura la responsabilité de traiter les requêtes HTTP GET pour l'URI **/contacts**.

```
app.get('/contacts', contact.list);
```

Nous verrons par la suite comment définir ce type de fonctions et implémenter le traitement des requêtes correspondantes.

La dernière chose à faire consiste à mettre en place un serveur afin de mettre à disposition l'application via le protocole HTTP. Pour ce faire, nous allons utiliser directement le support HTTP de Node et sa fonction **createServer**. Notre application Express sera directement en paramètre de cette fonction et le démarrage du serveur est effectif lors de l'appel de sa méthode **listen**. Le code suivant décrit la mise en place du serveur au sein du fichier **app.js** :

```
http.createServer(app).listen(app.get('port'), function(){
  console.log("Express server listening on port " + app.get('port'));
});
```

Maintenant que les fondations de notre application sont posées, il est temps d'implémenter ces traitements. La figure en bas de page donne une vue générale de l'organisation de l'application et de ses différents constituants [Fig.2].

> Implémentation des contrôleurs

Nous commençons par un contrôleur. Ce dernier correspond à la fonction passée en paramètre lors de la définition des routes. Elle accepte en paramètre les objets **request** et **response** relatifs à la requête courante de l'utilisateur, respectivement une représentation objet de la requête et de la réponse. Cette fonction a la responsabilité de traiter la requête, récupérer éventuellement des données et passer ensuite la main à la vue. Ce dernier traitement se réalise en se basant sur la méthode **render** de l'objet **response** précédent. Il prend en paramètre l'identifiant de la vue à utiliser ainsi qu'un tableau associatif correspondant aux données. L'exemple ci-dessous montre comment implémenter un contrôleur Express utilisant des données de contacts en dur puis passe la main à la vue **contacts** :

```
exports.list = function(request, response) {
  var contacts = [
    { id: 1, firstName: "Ryan", lastName: "Dahl" },
    { id: 2, firstName: "Tj", lastName: "Holowaychuk" }
  ];
  response.render('contacts', { contacts: contacts, title: 'Liste des contacts' });
};
```

L'exemple précédent est très simple et ne fait pas appel à une base de données. Si nous l'adaptions, il convient de bien faire attention aux aspects asynchrones éventuels lors d'appels aux bases de données. En effet, l'appel du rendu de la vue doit bien se trouver au sein de la

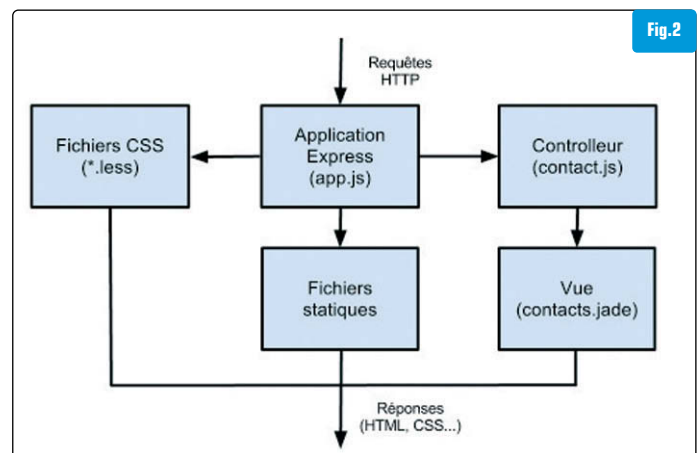


Fig.2

fonction recevant les résultats. Le code suivant illustre l'adaptation utilisant une base PostgreSQL en s'appuyant sur le client **pg** pour cette base (<https://github.com/brianc/node-postgres>) :

```
exports.list = function(request, response) {
  pg.connect(connectionString, function(err, client) {
    client.query("select id, firstName, lastName from contacts", function(err, result) {
      res.render('users', { contacts: result.rows });
    });
  });
};
```

Note : Une bonne pratique lors de l'écriture d'applications pour Node est de réduire au maximum le niveau d'imbrication des fonctions de rappel, qui peut vite augmenter de par l'approche asynchrone. Par exemple, pour le code précédent, il serait intéressant de scinder les traitements entre les appels à la base et ceux du contrôleur.

```
exports.list = function(req, res) {
  getContacts(function(err, contacts) {
    res.render('contacts', { contacts: result.rows });
  });
}

function getContacts(callback) {
  pg.connect(connectionString, function(err, client) {
    client.query("select id, firstName, lastName from contacts", function(err, result) {
      callback(err, err ? null : result.rows);
    });
  });
}
```

> Implémentation des vues avec Jade et Less

Nous arrivons à la dernière section de notre article, l'implémentation des vues afin de présenter dans une page HTML la liste des contacts récupérés. HTML, étant dérivé de XML, est un langage très verbeux. Il faut ouvrir et bien fermer les balises, et décrire les attributs sans oublier de fermer les guillemets. Le document HTML doit être bien formé pour obtenir le rendu souhaité. Jade (<http://jade-lang.com/>) est un moteur de templates qui met à disposition une approche visant à simplifier cela principalement en se basant sur l'indentation des lignes et des raccourcis d'écriture. Le code suivant illustre un exemple simple de template Jade :

```
doctype 5
html
  head
    title= title
    link(rel='stylesheet', href='/stylesheets/style.css')
  body
    p Hello!
```

Indenter une ligne correspond à ajouter un niveau d'imbrication. Ainsi la balise **head** résultante sera bien incluse dans la balise **html**. Vous pouvez également remarquer la manière de définir le contenu d'une balise à partir d'un élément présent dans le modèle au niveau du titre avec le symbole **=**. Avant de continuer, regardons rapidement ce que donne ce code après transformation par Jade :

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello</title>
    <link rel="stylesheet" href="/stylesheets/style.css">
  </head>
  <body>
    <p>Hello!</p>
  </body>
</html>
```

Nous venons de décrire un exemple simple de template Jade mais nous souhaitons désormais présenter les contacts que nous avons récupérés dans notre contrôleur. Pour ce faire, il convient de réaliser une boucle sur la liste correspondante. Ceci est possible avec Jade par l'intermédiaire de l'élément **each**. Ce dernier prend en paramètres le nom de la variable correspondant au tableau dans le modèle, comme le montre le code suivant :

```
h1= title
ul
  each contact in contacts
    li #{contact.id} - #{contact.firstName} #{contact.lastName}
```

Jade supporte également la possibilité de mettre en place des layout. Ces derniers permettent de modulariser et réutiliser les parties récurrentes de l'interface graphique (en-têtes, pieds de page...). Pour ce faire, il suffit de créer plusieurs fichiers Jade, un pour le layout et un par page. Le lien entre la page et le layout se réalise par l'intermédiaire d'un héritage et d'une référence.

Commençons par le layout. Il s'agit d'un fichier Jade classique si ce n'est la présence d'un élément **block**. Cet élément permet d'embarquer un bloc de contenu dont le nom est **content** ici. Ce bloc sera substitué par la suite lors du rattachement à une page.

```
doctype 5
html
  head
    title= title
    link(rel='stylesheet', href='/stylesheets/style.css')
  body
    block content
```

Dans notre page, il suffit d'étendre le template avec un élément dénommé **extends** en précisant le nom du template du layout, ici **layout** car nous l'avons stocké dans un fichier **layout.jade**. La seconde étape consiste à définir le bloc de contenu dénommé **content**. Ce bloc sera alors placé dans le layout à la place de l'élément **block** correspondant.

```
extends layout

block content
  h1= title
  ul
    each contact in contacts
      li #{contact.id} - #{contact.firstName} #{contact.lastName}
```

Comme nous l'avons évoqué précédemment, Express intègre un support de Less. Ce dernier permet de convertir à la volée les feuilles de styles Less en CSS lorsqu'elles sont demandées. Il suffit uniquement de créer un fichier portant l'extension **.less** qui sera utilisé pour générer un fichier **.css** correspondant. Par exemple, si le fichier **styles.css** est demandé via HTTP, le fichier **styles.less** sera utilisé afin de le créer. Nous ne nous attarderons pas plus sur Less ici mais sachez que l'outil supporte les expressions CSS et fournit des facilités d'écriture afin de garantir un même rendu pour les différents navigateurs sur des fonctionnalités telles que les dégradés et les bordures arrondies.

> En résumé

Dans cet article, nous avons décrit comment implémenter une application Web pour Node en utilisant Express, le plus populaire et utilisé des frameworks Web pour Node. Ce framework met en œuvre le patron de concept MVC (Model View Controller) afin de bien séparer les responsabilités de traitement des requêtes Web de la présentation des données manipulées. Nous avons vu toute la puissance de l'outil puisqu'en très peu de lignes de code, nous arrivons à mettre en place une application Web bien structurée et maintenable, d'affichage de contacts en utilisant JavaScript et basée sur Node. De plus, Express permet la création simplement et rapidement d'un squelette d'applications Web pour débiter le développement. Express fournit de plus un ensemble de fonctionnalités configurables afin d'enrichir les traitements de l'application (compression des données, sécurité...). Après configuration, ils prennent part de manière transparente dans la chaîne de traitement des requêtes. Il est à noter qu'Express permet l'activation de ces fonctionnalités selon l'environnement d'exécution.

Thierry Templier (thierry.templier@restlet.com)
Architecte R&D, Restlet SAS
Développeur du portage de Restlet en JavaScript pour Node
(<https://github.com/restlet/restlet-framework-js>)

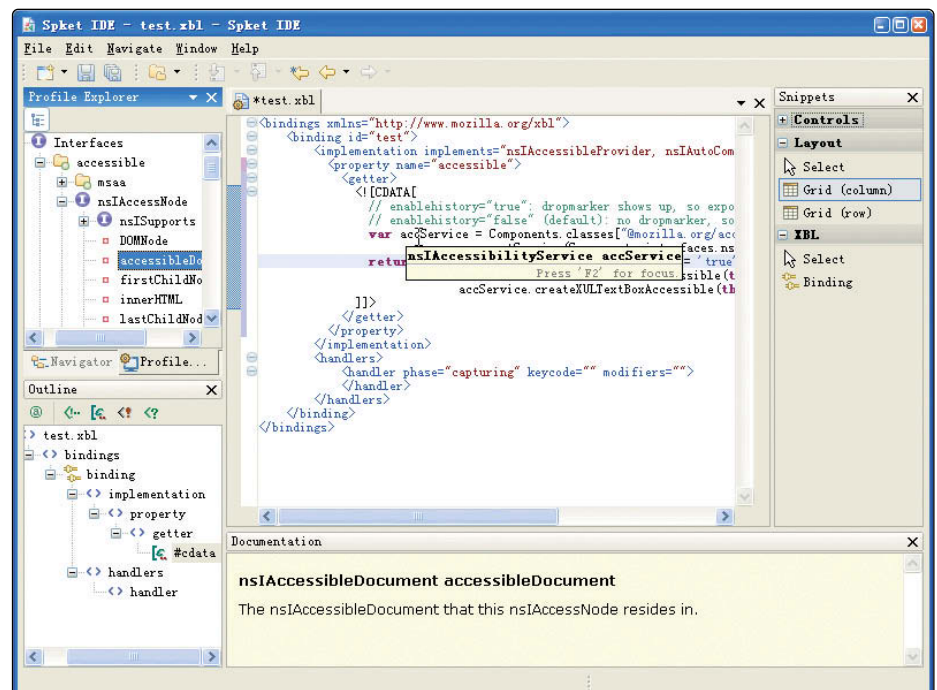
JavaScript/développeur : l'Amour avec un grand A.

Aujourd'hui au centre de toutes les attentions, le JavaScript n'a pas toujours été aussi plébiscité. Mal aimé il y a encore quelques années, il est rapidement devenu le langage le plus en vogue chez les développeurs d'applications Web. Comment expliquer ce succès fulgurant, quels sont les frameworks à regarder de près et à quoi faut-il s'attendre dans le futur... ? Décryptage.

Vilain petit canard du Web dans les années 2000, le JavaScript s'est aujourd'hui taillé une place de choix dans la boîte à outils des développeurs. Le motif de ce désamour initial ? Un langage difficilement abordable et entouré de beaucoup d'a priori. Redouté des développeurs, le JavaScript a souvent fait figure de mauvais élève à l'école du Web. Le JavaScript serait-il réservé à l'élite ? Sous-entendu démenti en quelques années grâce aux évolutions qui s'enchaînent : introduction de l'AJAX, et émergence des frameworks que l'on connaît aujourd'hui (jQuery, Mootools, Scriptaculous, ExtJS ou encore Sencha) qui envahissent le Web (à l'heure actuelle, jQuery serait présent sur près d'un site sur deux). Une revanche inattendue, pour autant, la transition entre un JavaScript distillé au compte-gouttes dans des pages HTML et la propulsion d'applications Web dynamiques ne s'est pas faite en un jour. Si les bibliothèques ont participé à l'accélération du processus, c'est tout l'écosystème du Web qui a grandi simultanément.

> Un langage en pleine mutation

En 2005, l'AJAX fait son apparition ouvrant de nouveaux horizons aux développeurs. Des applications d'un nouveau genre déboulent sur le marché, proposant une expérience utilisateur inédite. Cette déferlante d'applications, en rupture avec tout ce qui se faisait précédemment, s'inscrit dans la vague «Web 2.0». GMail est considéré comme l'un des pionniers de l'AJAX, ayant rendu populaire le XMLHttpRequest. Les évolutions du langage s'enchaînent, décuplant les possibilités offertes aux développeurs. Amélioration des performances, puissance de calcul accrue, introduction du JSON, les fonctionnalités natives du JavaScript ne cessent de s'étoffer. Le JavaScript est aujourd'hui deve-



nu un allié indispensable dans la conception d'une application Web. Pour rester dans la course, les navigateurs ont dû s'adapter. Le moteur JavaScript est même devenu un argument marketing pour attirer les développeurs. En 2008, Firefox ouvre les hostilités et annonçant «des performances entre 20 et 40 fois supérieures» avec Tracemonkey. Google répondra avec V8, la guerre était déclarée.

Dans une bataille qui se joue entre benchmarks d'un côté et expérimentations Web de l'autre, il est difficile de désigner un gagnant. Un seul constat s'impose : le JavaScript d'aujourd'hui n'est plus ce langage relégué au second plan, car trop peu performant et hasardeux.

L'union faisant la force, le couplage des fonctionnalités natives de JavaScript et du HTML5 permettent maintenant de rivaliser avec les logiciels applicatifs. Travail hors-

ligne grâce au stockage sur le disque de l'internaute, Web sockets, push, modélisation via le canvas... Les perspectives semblent infinies.

> L'âge d'or du JavaScript

Il y a encore quelques années, la simple évolution du nom «JavaScript» aurait fait «tiquer» n'importe quel développeur Web ; associant le langage à des mauvaises pratiques de codage, une interprétation aléatoire suivant les navigateurs et l'assurance de maux de tête en fin de journée. L'arrivée des frameworks a radicalement changé la donne et a contribué à la réduction des temps de développement des scripts front-end. Le principe : proposer des dizaines de méthodes simples à implanter afin de simplifier l'écriture de code. Grâce à elles, il est devenu plus simple de développer des sites plus complexes ! En effet, si le JavaScript est un langage cross-

Comment choisir son framework ?

Il existe plusieurs «grandes familles» de frameworks :

- les frameworks permettant de travailler avec le DOM. Ce sont les plus répandus : JQuery, Dojo, YUI...
- ceux permettant d'appliquer une architecture MVC : Backbone.js est le plus populaire.
- d'autres permettant de gérer du templating comme Mustache, Handlebar ou Underscore
- ou encore des librairies de détection de type Modernizer, aidant au progressive enhancement (désactivation de fonctionnalités si elles ne sont pas supportées par le navigateur).

Cette liste est non exhaustive et uniquement présente dans le but de structurer les choses. Il est également possible de les faire évoluer les uns avec les autres. Il faut garder à l'esprit qu'aucun framework n'est meilleur qu'un autre. Ils fournissent tous des solutions répondant à un enjeu. Avant de vous engager avec un, mettez en place une liste de vos contraintes : techniques, délais, budget.

Puis posez-vous les bonnes questions :

- Le framework supporte-t-il les navigateurs dans ma cible ?
- Si vos deadlines sont serrées et que vous ne le connaissez pas, essayez d'évaluer relativement le temps d'apprentissage d'un framework.
- La librairie inclut-elle les éléments dont j'ai absolument besoin ? Les autres sont-ils retirables ?
- Est-il simple d'ajouter un module ? De debugger mon code ?
- Que valent les performances par rapport aux concurrents ?
- La documentation est-elle suffisamment fournie et claire ? Quid de la communauté ?
- À quand remonte la dernière mise à jour ?
- Quelle est la licence ? Est-elle applicable à mon projet ?

langage quand il est utilisé habilement. L'avenir nous promet encore de belles surprises et il n'est pas exclu de voir d'autres utilisations innovantes du langage dans les prochains mois.

Matthieu Loussinian

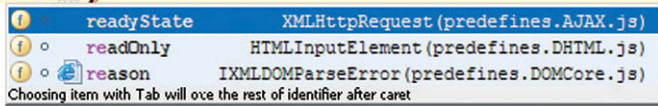
Directeur général de FrenchTouched, société spécialisée en développement & consulting e-commerce.

<http://www.frenchtouched.fr>

```

38
39 function updateElement(elementId) {
40   var xmlHttp = createXmlHttp();
41   if (xmlHttp) {
42     xmlHttp.onreadystatechange = function () {
43       if (xmlHttp.readyState == 4) {
44         // ...
45       }
46     }
47   }

```



platform et qu'il fait l'objet de normes, il n'en reste pas moins une légère différence d'interprétation entre navigateurs. Seulement, ces quelques différences mineures, peuvent rendre caduque l'intégralité d'une Web app. Alors qu'il était nécessaire de maîtriser toutes ces subtilités il y a encore peu, les frameworks soulagent le travail des développeurs en palliant ces différences, et en assurant une interopérabilité complète des codes. La liberté est ainsi laissée aux développeurs de se concentrer sur leur algorithme.

> L'apparition de nouveaux outils

En parallèle, des outils de débogage font leur apparition, tel Firebug, jouissant d'une grande renommée sur Firefox, ou encore JSLint, service en ligne permettant de pointer les problèmes potentiels de code (erreurs de syntaxe, respect des conventions d'écriture, optimisation du code). Tous les navigateurs intègrent maintenant leurs propres outils de développement, incluant une console JavaScript. La facilité à écrire et debugger du

ment, il faudra passer par l'étape du refactoring : modifier la structure du code en gardant un fonctionnement identique. Ce refactoring est une étape intéressante dans le sens où elle permet d'améliorer l'architecture d'une app et éventuellement les performances.

Là encore, pour aider les développeurs dans cette tâche, plusieurs frameworks ont vu le jour. QUnit est le plus populaire et s'insère parfaitement avec JQuery tandis que FireUnit ajoute un onglet de test dans la console de Firebug.

Tous ces frameworks JavaScript ont également ouvert la porte à des possibilités jusqu'alors peu explorées : animations riches, gestion de sélecteurs complexes. Un vrai plus pour l'expérience utilisateur. Toutes ces librairies tirent parti de l'orienté objet et sont donc extensibles à souhait. Grâce au partage des ressources sur Internet, de nombreux modules sont généreusement mis en téléchargement (slideshow, fenêtres modales, scripts de validation côté utilisateur, ...) et contribuent à rendre le langage accessible. Node.JS est l'autre révélation. Très en vogue sur les sites Web à forte audience, Node.JS a été conçu pour permettre de bâtir rapidement des applications scalables. La plateforme s'appuie sur le moteur

V8 de Chrome pour exécuter du JavaScript côté serveur. Ainsi, le même langage est utilisé pour le développement côté client & serveur. Il est également important de rappeler une démarche en pleine expansion visant à améliorer l'expérience utilisateur (UX). En lien avec le HTML et le CSS, il est possible de créer une vraie expérience dynamique et interactive entre l'internaute et l'application Web (ou le site vitrine). Cela passe également par des petites améliorations moins remarquées, mais, tout aussi importantes, comme les retours d'informations, l'exécution de requêtes asynchrones, permettant d'accélérer les processus d'envoi et récupération d'informations en AJAX de façon non bloquante. Le JavaScript est un formidable

“ Il est devenu plus simple de développer des sites plus complexes ! ”

code conditionnant pour beaucoup l'adoption d'un langage, ces outils ont contribué à attirer les développeurs vers le langage.

Enfin, en plus de debugger son application, le développeur sait qu'il est important de s'assurer du bon fonctionnement de chaque module en effectuant des tests unitaires. En JavaScript, l'opération peut sembler laborieuse. En effet, il est fréquent de se retrouver avec divers modules répartis sur des pages différentes, le tout mélangé dans des bouts de HTML, quand on ne trouve pas des appels inline... Rajoutez à cela des portions de code attachées à des événements (click, hover, survol, ...) et vous obtiendrez une belle soupe indigeste. Si les tests unitaires n'ont pas été prévus en amont du développe-

JavaScript est souvent incompris !



S'il y a un langage dont pratiquement tout le monde a entendu parler, que l'on travaille dans l'informatique ou non, c'est bien le JavaScript. Et pourtant il reste souvent incompris. Les raisons en sont multiples : son nom n'a jamais aidé à y voir clair et sa réussite et sa vulgarisation font que chacun a souvent une idée différente de ses avantages et de ses inconvénients.

Cela commence par un problème de nommage. Conçu en 1995 à Netscape, il devait servir d'alternative au langage Java, pour permettre notamment aux développeurs novices ou non professionnels d'écrire des scripts pour les pages s'exécutant dans le célèbre navigateur de la compagnie. Ce statut lui a valu son nom : JavaScript. Or, le nouveau langage n'hérite de son prédécesseur que le nom. Certes, sa syntaxe ressemble à celle du langage développé par Sun, et donc aussi au C ou au C++, mais les similitudes s'arrêtent là. Ses caractéristiques de langage fonctionnel le rapprochent plus du Lisp ou du Scheme, les parenthèses en moins.

> Une capacité d'évoluer

Dès sa sortie, JavaScript est rapidement adopté par les développeurs de pages web, qui trouvent dans le langage un moyen simple et efficace d'ajouter de l'interactivité au code HTML statique, ce qui a poussé les concurrents de Netscape, et notamment Microsoft, à écrire leurs propres implémentations pour leurs navigateurs. L'accueil chaleureux des développeurs et des constructeurs pousse Netscape à soumettre la définition du langage à un organisme de certification, la *European Computer Manufacturers Association* ou ECMA. C'est ainsi que, seulement 2 ans après sa création, JavaScript devient une implémentation d'un nouveau standard, l'ECMAScript. En quinze ans, le standard a peu évolué, mais son adoption massive est incontestable, malgré les difficultés qu'il a pu rencontrer au cours du temps. En effet, les implémentations du langage et des interpréteurs qui l'exécutent se sont longtemps caractérisés par leurs pauvres performances et leurs *bugs*. En dépit d'un manque de ressources pédagogiques de qualité, comme on en trouve abondamment pour

d'autres langages, JavaScript a été adopté par un public très large, et pas toujours professionnel, ce qui a aussi favorisé la prolifération de code de qualité médiocre, donnant par extension au langage une mauvaise réputation. Ce qui n'a, heureusement, pas empêché JavaScript de devenir l'un des plus importants moteurs derrière l'évolution technique du Web, en tout cas, côté client.

Au début des années 2000, c'est le J dans le nouveau terme « AJAX » qui décrit les techniques permettant aux sites web de réaliser des appels asynchrones sur le serveur et de proposer ainsi aux internautes des expériences plus riches et plus ergonomiques. Aujourd'hui, il est aussi pratiquement indissociable de celui qui commence à se faire une place importante dans le web, l'HTML 5, qui lui, expose ses API de drag-and-drop, de stockage, de géolocalisation, etc. permettant la création de sites extrêmement riches, incorporant multimédia et animations, et de plus en plus difficiles à différencier des applications natives.

Cette adoption du langage a contribué à l'apparition d'un écosystème très diversifié (et malheureusement un peu fragmenté) dans lequel des bibliothèques et des frameworks tels que jQuery, Prototype, Backbone, Knockout, et bien d'autres, s'efforcent d'accélérer et de faciliter le développement de sites web de plus en plus riches.

Mais à aucun moment il n'a été dit que JavaScript devait se limiter à l'exécution dans un navigateur web.

Dès la sortie du langage, Netscape proposait une implémentation serveur de celui-ci et pendant longtemps le JScript, l'implémentation de Microsoft, était une alternative au VBScript pour écrire des *Active Server Pages* qui, comme leur nom l'indique, s'exécutent côté serveur. Ces technologies n'existent plus aujourd'hui, mais depuis peu Node.js est venu combler le vide dans ce

domaine, proposant un environnement de développement serveur basé sur l'interpréteur JavaScript développé par Google pour le navigateur Chrome et conçu pour être extrêmement adaptable.

À différents moments dans son histoire, JavaScript a aussi réussi à quitter le Web pour s'installer confortablement sur le bureau de l'ordinateur personnel sous une multitude de formes. C'est lui, par exemple, qui anime les *widgets* du Dashboard Mac OS X ou les *gadgets* sous Windows. C'est aussi JavaScript qui est utilisé dans le développement des extensions des navigateurs comme Firefox ou Chrome. Et au cas où on croirait que JavaScript ne sert à développer que des « accessoires », Microsoft en a fait un langage de première classe sur PC avec la sortie de Windows 8, puisque le couple HTML5/JavaScript permet d'écrire des applications complètes capables d'utiliser toute la puissance de la plateforme.

Il est donc impossible de nier que ses qualités techniques et sa portabilité ont permis à JavaScript de s'imposer aujourd'hui dans tous les domaines. Soutenu par des géants de l'industrie, son écosystème gagne en robustesse et commence à être considéré sérieusement par les développeurs professionnels. Ceci résulte en une augmentation des ressources et du code de qualité à la disposition des nouveaux arrivants, qui en profiteront pour ajouter à leur répertoire un outil promettant de nombreuses opportunités professionnelles.

Mauricio Diaz Orlich - Consultant - Wygwm

A lire dans le prochain numéro

N°160-Février 2013, à paraître le 31 janvier 2013

• Développer pour Facebook

La plateforme, les API et SDK, opengraph, le modèle économique

• Cloud, open Cloud :

ce qui change pour les développeurs

Ecrire un module d'extension à Prestashop

Prestashop est un CMS orienté commerce en ligne. Dès son installation il s'avère très complet et très efficace. De nombreux modules existent qui peuvent l'enrichir, mais tôt ou tard, il sera nécessaire d'écrire soi-même un module d'extension pour des besoins particuliers.

Prestashop est un logiciel pour le commerce en ligne de même nom que la société qui l'édite. Il est open source, sous licence Open Software 3.0. A l'origine, il s'appelait phpOpenStore, et il a été créé en 2005 par 5 étudiants de l'école d'informatique Epitech. Très rapidement, il a connu le succès auprès de ceux qui ont besoin d'une solution rapide de mise en place d'un site de e-commerce. De par sa conception et son architecture, il mérite le titre de CMS. Mais un CMS pensé avant tout pour le commerce. D'entrée de jeu, la gestion des factures, des commandes est présente, alors qu'avec un CMS généraliste comme Drupal, vous devrez ajouter des modules. Par contre Prestashop n'est pas l'outil pour tenir un blog. De très nombreux modules d'extension sont disponibles pour Prestashop, et bien entendu ceux-ci sont essentiellement dédiés au commerce en ligne. Cependant, il viendra toujours le moment où il sera nécessaire de personnaliser rapidement Prestashop pour un besoin particulier. Nous allons faire le tour de ce qu'il faut savoir pour être opérationnel rapidement. Pour cet article, nous avons utilisé Prestashop 1.4.9.

1 INSTALLATION DE PRESTASHOP

Comme WordPress que nous avons étudié dernièrement, Prestashop est conçu pour fonctionner dans un hébergement bas de gamme, à condition que celui-ci vous fournisse PHP 5 et MySQL 5 au minimum. La procédure d'installation est des plus simples. Tout d'abord vous devez disposer d'une base de données, ce que vous fournira un hébergement bas de gamme, ou que vous mettrez en place sur un serveur dédié. Ensuite, on commence par télécharger l'archive de Prestashop. Puis, si vous disposez d'un serveur dédié, vous uploadez l'archive et la décompressez sur place, sinon vous décompressez l'archive sur votre machine de travail, puis vous uploadez tout le contenu décompressé sur votre serveur, via un client FTP. Ceci fait, vous allez visiter votre site sous Prestashop. Ce dernier se rendra compte qu'il n'est pas installé et lancera la procédure. Vous aurez quelques renseignements à donner, dont le nom de la base de données à votre disposition, et l'installation se fera entièrement. Au cours de la procédure, on vous demandera si vous souhaitez une installation simple, ou une installation plus lourde comprenant tout un jeu de modules. Sans hésiter, optez pour ce dernier choix. Une fois l'installation terminée, vous devrez supprimer le répertoire 'install' de l'arborescence de Prestashop et aussi, pour des raisons de sécurité, renommer le répertoire admin d'un nom qui vous conviendra et, si possible, que vous seul connaîtrez. Ensuite, pour accéder à l'interface d'administration de votre site sous Prestashop, vous donnerez cette url dans un navigateur :

http://www.mon-domaine.com/nouveau_repertoire_admin

2 L'ARCHITECTURE DE PRESTASHOP

D'entrée de jeu, vous vous retrouvez avec un site dont la page d'accueil est très complète et structurée [Fig.1]. La page est construite



sur trois colonnes, avec en-tête et pied de page. Les colonnes sont faites d'un empilage de pavés ou blocks. Un module d'extension peut agir sur toutes ces parties. Prestashop s'appuie sur la programmation objet en PHP, et sur une architecture 3-tiers dans l'esprit, nous dit sa documentation, de l'architecture MVC, ou Modèle-Vue-Contrôleur, et organisée comme ceci, toujours selon la documentation :

- **Object/données.** L'accès à la base de données se fait par le biais des fichiers du dossier /classes.
- **Contrôle des données.** Le contenu envoyé à l'utilisateur est vérifié par les fichiers du dossier racine.
- **Apparence.** Tous les fichiers du thème sont dans le dossier /themes. Les dossiers dont il est question sont ceux de l'arborescence de Prestashop. En jetant un oeil sur eux, on trouvera bien des renseignements qui devraient figurer dans la documentation, mais qui n'y étaient pas au moment de la rédaction de cet article. Il est vrai que la documentation était signalée comme étant en cours de réécriture. Par exemple, en visitant ce dossier 'classes', vous verrez que Prestashop propose une fine couche d'abstraction pour le travail avec les bases de données. Ainsi, consultez le fichier Db.php et si vous utilisez MySQL, le fichier MySQL.php, qui implémente la classe abstraite DbCore de Db.php. Dans cette dernière vous trouverez des méthodes fort utiles, telles que pSQL qui permet d'assainir des données devant être passées à des requêtes SQL et de se prémunir ainsi des injections. Contrairement à ce que dit la documentation, avec les versions récentes de Prestashop, le contrôle des données est effectué, par les fichiers rangés dans le répertoire 'controllers'. Les fichiers à la racine ne font plus que renvoyer vers ce dossier.

3 UN PREMIER MODULE

Commençons par créer un premier module en nous appuyant sur le Guide du Développeur qui accompagne Prestashop, et en le complétant lorsque cela est nécessaire. Un module Prestashop est constitué d'un sous-dossier du dossier 'modules' de Prestashop.

Ensuite ce sous-dossier doit contenir au moins un fichier .php de même nom que le sous-dossier. Pour notre exemple nous avons créé un dossier nommé basicprog, que vous trouverez sur notre site. Voici le contenu de notre fichier basicprog.php :

```
<?php

if ( !defined( ' _PS_VERSION_ ' ) )
    exit;

class BasicProg extends Module
{
    public function __construct()
    {
        $this->name = 'basicprog';
        $this->version = 1.0;
        $this->author = 'fred';
        $this->need_instance = 0;

        parent::__construct();

        $this->confirmUninstall = $this->l('Do you really want to
uninstall BasicProg?');
        $this->displayName = $this->l('Basic Prog');
        $this->description = $this->l('A very basic demo');
    }

    public function hookLeftColumn($params)
    {
        global $smarty;
        return $this->display(__FILE__, 'basicprog.tpl');
    }

    public function install()
```

```
{
    if(parent::install() == false)
        return false;
    if(!$this->registerHook('leftColumn'))
        return false;
    if(!$this->registerHook('rightColumn'))
        return false;
    return true;
}

function uninstall()
{
    if(!parent::uninstall())
        return false;
    return true;
}
}>
```

Le rôle des deux premières lignes de code est d'éviter l'exécution du contenu du fichier .php hors contexte, par une simple requête sur son URL. Nous voyons ensuite une classe qui dérive de la classe module. Par convention, le nom de cette classe doit être le même que celui du module, en notation CamelCase. Nous initialisons ensuite quelques propriétés publiques de la classe. Parmi elles, *name* doit être le nom du module. La propriété *need_instance* précise si le code de notre module doit être chargé ou non lorsqu'on navigue dans l'interface d'administration de Prestashop. Après l'appel au constructeur de la classe parent, viennent de nouvelles propriétés publiques sous la forme de chaînes de caractères. La première est le message de confirmation émis par le backoffice de Prestashop. Les autres concernent l'affichage du module dans l'interface du backoffice. Dans le code, nous donnons du texte en anglais, mais celui-ci s'affiche en français dans l'interface d'administration, ainsi que nous le voyons [Fig.2].

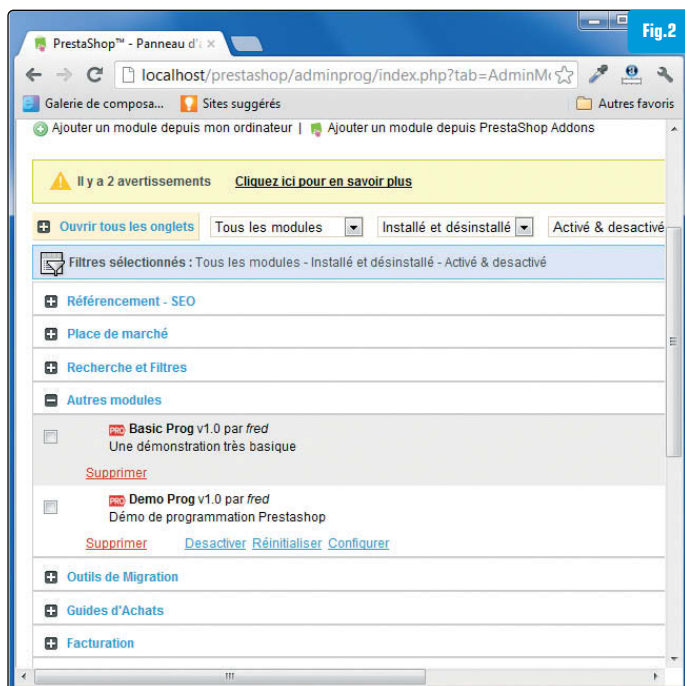
Ceci est la façon normale de procéder avec Prestahop qui est conçu dès le départ pour l'internationalisation. Bien remarquer dans notre code que la traduction des chaînes est préparée par un appel à la méthode 1 de la classe module. Toutes les chaînes ainsi répertoriées apparaissent dans l'interface de traduction du backoffice (onglet Outil, puis menu Traductions), pour les langages activés. Il suffit alors de donner les chaînes traduites pour que le module soit internationalisé :) Les chaînes de traduction sont déposées par Prestahop dans le répertoire du module, dans un fichier nommé d'après la langue, fr.php dans notre cas.

Ce fichier ne doit pas être modifié manuellement. Notre code pourrait s'arrêter au constructeur de notre classe. Nous aurions un module qui ne ferait rien, mais qui serait parfaitement fonctionnel. Nous pourrions l'installer depuis le backoffice, ce qui produirait un fichier XML supplémentaire dans le répertoire de notre module. Là encore ce fichier ne doit pas être modifié manuellement.

Attention au lien 'supprimer' dans le backoffice modules, qui ne doit pas être confondu avec le lien *désinstaller*.

Le lien 'supprimer' efface purement et simplement le répertoire du module et son contenu.

Attention à ne pas détruire ainsi par maladresse tout votre code pendant le développement ! Il est maintenant temps que notre module fasse un petit quelque chose.



Nos chaînes de textes apparaissent traduites dans le backoffice.

4 LES HOOKS

Comme beaucoup de CMS, Prestashop fournit un mécanisme de hooks. Un hook est un point d'accrochage pour des fonctions enregistrées auprès du système. Ces fonctions sont invoquées au moment opportun par le système, ou par des modules. Il existe des hooks pour tout : panier, produit, paiement affichage, etc. La liste, plus ou moins à jour, est donnée dans le guide du développeur. Ne manquez toutefois pas de visiter la table `ps_hook` en base de données. L'endroit naturel pour enregistrer un hook est la méthode `install` de la classe du module. C'est ce que fait notre code, qui enregistre un hook pour afficher à la volée un block dans la colonne gauche des pages de notre site. Le hook lui-même génère l'affichage du block en invoquant un template Smarty, qui est le moteur de template de Prestashop. Attention, les hooks doivent obéir eux aussi à une convention de nommage, qui est donnée dans le guide du développeur. Voici notre template Smarty, d'abord dans une forme rudimentaire. (Le lecteur trouvera les sources complets, y compris le fichier `basicprog-page.php`) :

```
<!-- Block Left Basic Prog -->
<div id="basic-prog-left-bloc" class="block left-block">
<h4>Programmez!</h4>
<div class="block-content">
<ul>
<li><a href="{ $base_dir }modules/basicprog/basicprog-page.php" title="Cliquez ici !">Abonnez-vous!</a></li>
</ul>
</div>
</div>
<!-- /Block Left Basic Prog -->
```

Mais dans l'esprit de Prestashop, il convient de prévoir dès le départ l'internationalisation, y compris dans les templates. On donnera donc :

```
<!-- Block Left Basic Prog -->
<div id="basic-prog-left-bloc" class="block left-block">
<h4>Programmez!</h4>
<div class="block-content">
<ul>
<li><a href="{ $base_dir }modules/basicprog/basicprog-page.php" title="{ l s='Click here !' mod='basicprog' }">{ l s='Subscribe!' mod='basicprog' }</a></li>
</ul>
</div>
</div>
<!-- /Block Left Basic Prog -->
```

Dès le premier affichage dans le navigateur, Prestashop place les templates dans un cache pour des questions de performance. Au cours du développement, désactivez le cache, sinon vous ne verrez pas le résultat de vos modifications dans les templates. Pour l'instant, notre template casse la structure des pages de Prestashop, ce que nous résoudrons plus loin [Fig.3]. Au départ, notre block apparaîtra très probablement en bas de colonne. Ce problème se règle par l'administrateur du site, via le backoffice, ainsi que les exceptions d'affichage, lorsque le block ne doit pas apparaître sur telle ou telle page. Il est envisageable d'agir sur l'ordre des blocks par program-

mation en modifiant la table `ps_hook_module` lors de l'installation du module. Néanmoins on doit auparavant bien se poser la question de la pertinence de faire cela. Car si de nombreux modules font de même, cela perdra vite du sens.

5 AJOUTER UN NOUVEAU HOOK

Les hooks préexistants sont a priori suffisants pour créer des modules simples, mais il n'est pas interdit de voir grand :) Ainsi on peut imaginer le projet d'un super module ambitieux et complexe qui pourrait être enrichi de sous-modules. Ces sous-modules utiliseraient des hooks proposés par ce super module. Comment, dans ce cas, mettre en place un hook ? D'abord on créera une entrée dans la table `ps_hook`. Ensuite, le code de notre super module invoquera les fonctions enregistrées comme ceci :

```
Module::hookExec('le-nom-de-notre hook');
```

Ainsi toutes les fonctions enregistrées seront invoquées, mais dans un ordre a priori indéterminé.

6 UN MODULE COMPLET

Nous allons maintenant passer à notre second exemple qui est un module complet, et configurable depuis le backoffice. Notre module crée à nouveau un block en colonne de gauche, et un clic sur ce lien simule l'achat d'un abonnement à votre magazine préféré :) La configuration dans le backoffice permet de tricher, en doublant le nombre des abonnés :) Voici les méthodes importantes de la classe de ce nouveau module (DemoProg sur notre site)

```
public function install()
{
    $this->initDB();
    if(parent::install() == false)
        return false;
    if(!$this->registerHook('leftColumn'))
        return false;
    if(!Configuration::updateValue('DEMOPROG_DOUBLER_ABOS', 0))
        return false;
    return true;
}

function uninstall()
{
    Db::getInstance()->Execute('DROP TABLE `'. _DB_PREFIX_ . 'demoprogram`');
    if(!parent::uninstall())
        return false;
    if(!Configuration::deleteByName('DEMOPROG_DOUBLER_ABOS'))
        return false;
    return true;
}
```



Pour l'instant notre template d'affichage ne s'intègre pas à Prestashop.


```
private function _displayForm()
{
    $this->_html .=
    '<form action='.$_SERVER['REQUEST_URI'].' method=post>'
    <div style=»margin:0 0 20px 0;»>
        <input type=»checkbox» name=»DEMOPROG_DOUBLER_ABOS» id=
        »DEMOPROG_DOUBLER_ABOS»
        style=»vertical-align: middle;» value=»1» '
        .(Configuration::get('DEMOPROG_DOUBLER_ABOS') ? 'checked'
        =»checked' : '').' />
        <label class=»t» for=»DEMOPROG_DOUBLER_ABOS»>'
        . $this->l('Double subscriptions').'</label>
    </div>
    <input name=»btnSubmit» class=»button» type=»submit»
    value=»Valider» />
</form>;
}

public function getContent()
{
    if (Tools::isSubmit('btnSubmit'))
    {
        Configuration::updateValue('DEMOPROG_DOUBLER_ABOS',
        (int)Tools::getValue('DEMOPROG_DOUBLER_ABOS'));
    }
    $this->_html .= '
    <fieldset>
    <legend>'. $this->l('Configuration Demo Prog').'</legend>;

    $this->_displayForm();

    $this->_html .= '</fieldset>;'
    return $this->_html;
}

public function initDB() {
    $id = 1;
    $nb_abo = 0;
    Db::getInstance()->Execute(
```

```
'CREATE TABLE IF NOT EXISTS `'._DB_PREFIX_.'demoprogram` (
`id` int(11) NOT NULL,
`nb_abos` int(11) NOT NULL
)ENGINE=MyISAM DEFAULT CHARSET=utf8;');

    Db::getInstance()->Execute('INSERT INTO `'._DB_PREFIX_.'demo
prog (id, nb_abos)
VALUES (1, 0)');
}
```

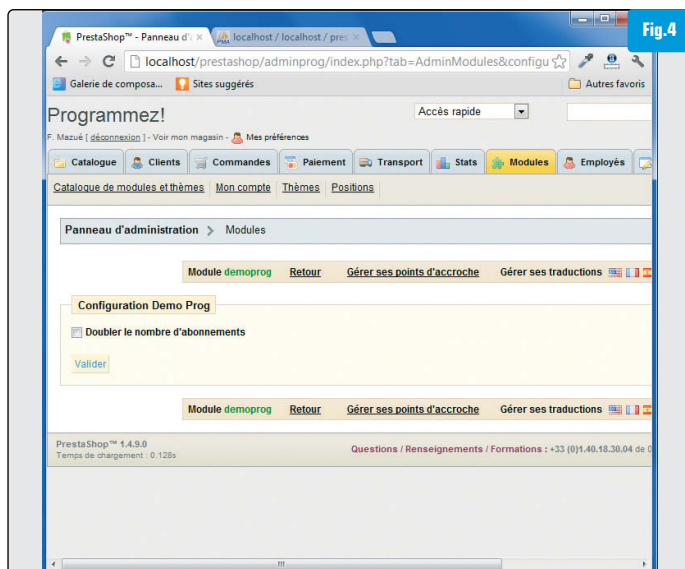
La méthode install commence par créer une table en base de données. Cette table ne contiendra qu'un enregistrement, pour conserver le nombre d'abonnés. Puis notre méthode install enregistre un hook, comme dans l'exemple précédent, et enfin définit une variable de configuration dont nous nous servons pour savoir si nous devons doubler le nombre d'abonnés. La question se pose alors de savoir comme faire apparaître cette variable de configuration dans le backoffice [Fig.4]. Le guide du développeur n'en dit rien ! C'est là qu'il est intéressant d'installer Prestashop avec de nombreux modules. Ainsi, j'ai trouvé la marche à suivre dans le code du module Google Site-map. La solution consiste à implémenter dans notre classe une méthode getContent qui fournit l'affichage pour le backoffice. Si l'affichage provient d'un clic sur le bouton de validation, on enregistre le nouvel état de la variable, et dans tous les cas, on affiche un formulaire. Vient ensuite la question de savoir comment obtenir un affichage intégré à Prestashop. Le lecteur trouvera la réponse dans le fichier demoprogram-page.php sur notre site, fichier qui sera invoqué lorsque l'utilisateur cliquera sur le lien du bloc:

Ce sont les includes de ce code qui assurent la structure de la page. Ensuite le point intéressant est la création, via \$smarty->assign, d'une variable qui sera accessible depuis le template qui est invoqué immédiatement après, et dont voici le code. (L'internationalisation est laissée au lecteur :-)

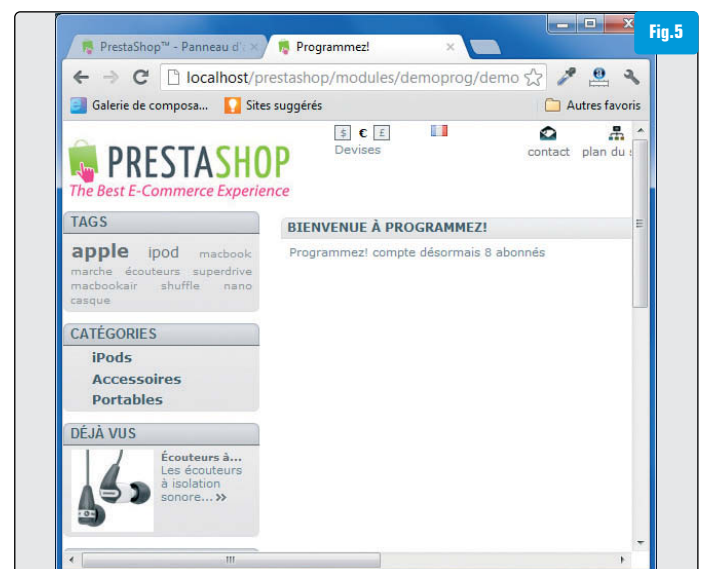
```
<h1>Bienvenue à Programmez!</h1>
<p>Programmez! compte désormais {$messageSmarty} abonnés</p>
```

Avec ce procédé, notre module produit un affichage bien intégré à Prestashop.

Frédéric Mazué - fmazue@programmez.com



Notre variable de configuration apparaît dans le backoffice.

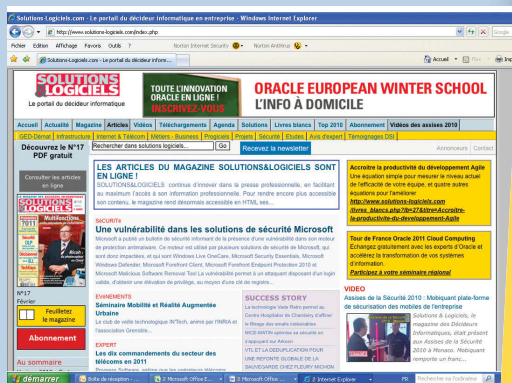


Cette fois notre module s'intègre correctement à Prestashop.

Les outils des Décideurs Informatiques

*Vous avez besoin d'info
sur des sujets
d'administration,
de sécurité, de progiciel,
de projets ?
Accédez directement
à l'information ciblée.*

Cas clients
Actu triée par secteur | **Avis d'Experts**



Actus / Événements | Newsletter | Vidéos

www.solutions-logiciels.com

☐ **OUI, je m'abonne** (écrire en lettres capitales)

Envoyer par la poste à : Solutions Logiciels, service Diffusion, GLIE - 17 chemin des Boulangers 78926 Yvelines cedex 9 - ou par fax : 01 55 56 70 20
1 an : 50€ au lieu de 60€, prix au numéro (Tarif France métropolitaine) - Autres destinations : CEE et Suisse : 60€ - Algérie, Maroc, Tunisie : 65€ , Canada : 80€ - Dom : 75€ Tom : 100€
10 numéros par an.

☐ M. ☐ Mme ☐ Mlle Société

Titre : Fonction : ☐ Directeur informatique ☐ Responsable informatique ☐ Chef de projet ☐ Admin ☐ Autre

NOM Prénom

N° rue

Complément

Code postal : Ville

Adresse mail

☐ Je joins mon règlement par chèque à l'ordre de SOLUTIONS LOGICIELS ☐ Je souhaite régler à réception de facture



Editeur : K-Now sarl au cap de 8000€
siège social : 21 rue de Fécamp, 75012 Paris

HTML5+jQuery

Pour tout Navigateur - Plateforme - Application - Appareil



IGNITEUITM
INFRAGISTICS JQUERY CONTROLS



Telecharger gratuitement la version d'essai
www.infragistics.com/igniteui-trial



Infragistics Sales France +33 (0) 800 667 307 • US 800 231 8588 • Europe +44 (0) 800 298 9055 • India +91 80 4151 8042 • APAC +61 3 9982 4545

Copyright 1996-2013 Infragistics Inc. Tous droits reserves. Infragistics et Infragistics sont enregistres comme etant marque deposee. Le logo est marque deposee Infragistics Inc. Toute autres marque deposee ou enregistree est deposee dans le respect de la propriete de celui qui la detient.